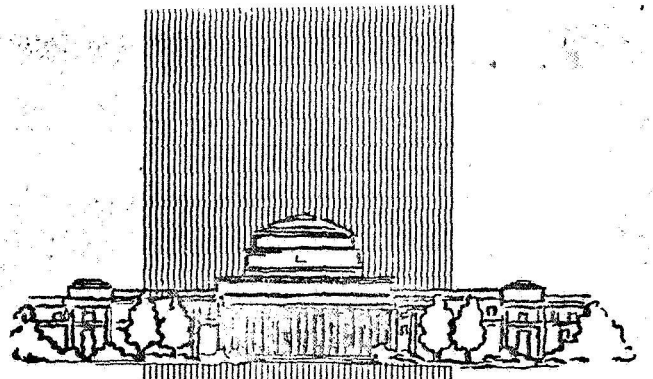
CR 151194



MASSACHUSETTS INSTITUTE OF TECHNOLOGY

CORRELATED MALFUNCTIONS IN REDUNDANT SYSTEMS

by

William Winiker Weinstein

September 1972

Degree of Master of Science

T-581

PREPARED AT

CHARLES STARK DRAPER LABORATORY

CAMBRIDGE, MASSACHUSETTS, 02139

CORRELATED MALFUNCTIONS IN REDUNDANT SYSTEMS

by

William Winiker Weinstein
S.B., Massachusetts Institute of Technology
(1968)

Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September, 1972

Signature of Author _____
Department of Aeronautics and Astronautics

Certified by _____
Thesis Supervisor

Accepted by _____
Chairman, Departmental Committee on Graduate Students

CORRELATED MALFUNCTIONS IN REDUNDANT SYSTEMS

by

William W. Weinstein

Submitted to the Department of Aeronautics and Astronautics, September 1972, in partial fulfillment of the requirements for the Degree of Master of Science.

ABSTRACT

This thesis considers the problem of correlated malfunctions in systems that employ redundancy to achieve high reliability. If malfunctions in corresponding simplex elements of a redundant system tend to be correlated, (a phenomenon referred to as systematic failure), the benefits of the redundancy will be negated. The actual reliability of the redundant system will be less than the figure derived by conventional redundancy analysis.

A model of erroneous behavior is developed around the premise that all malfunctions result from built-in system flaws that are activated by external events. In order to determine the potential correlations among flaws, a general structure for a system development timeline is constructed. Several methods for including the effects of correlations in the reliability equation are then explored.

It is concluded that correlated malfunctions represent a serious problem in the design and implementation of highly-redundant ultra-reliable systems, and that a reasonable estimation of the effects of correlated malfunctions can be made using the techniques developed.

Thesis Supervisor: Albert L. Hopkins, Jr.
Title: Associate Professor of Aeronautics and Astronautics

ACKNOWLEDGEMENT

The author would like to thank Dr. Albert L. Hopkins for the guidance and motivation he provided throughout the course of this thesis. He is especially grateful to Mr. Hugh Blair-Smith for the many stimulating discussions in which he offered suggestions and criticism.

This report was prepared under DSR Project 55-23890 sponsored by the Manned Spacecraft Center of the National Aeronautics and Space Administration through Contract NAS 9-4065.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

In recent years a great deal of effort has been expended in developing the theory and acquiring the practical experience needed to produce ultra-reliable electronic systems. The desire for ultra-reliable systems has always been present, but the technological capability to produce such systems has only recently become available.

The development of the technology to produce reliable electronic components was spurred largely by the aerospace industry. Many breakthroughs were a direct result of the requirements of the Apollo project, which was the first large-scale development program to place reliability above system cost. Current projects, such as the Space Shuttle, demand even higher reliabilities than those which were available for Apollo. In addition, other areas of endeavor are beginning to look towards ultra-reliable systems; air traffic control, automotive traffic control and biomedical instrumentation are but a few. These are real-time control systems in which loss of data or operational capacity due to a malfunction can lead to loss of life. They are unlike large-scale computing utilities where there are no life-critical operations and where an hour of down time may result in user inconvenience, but no threat to life.

Von Neumann's contention that reliable machines can be made by the appropriate interconnection of relatively unreliable machines[12] forms the basis for using redundancy to achieve ultra-reliability. From a practical viewpoint, however, it is necessary to start with a basically reliable unit, otherwise the degree of redundancy necessary to achieve ultra-reliability will produce a system that is unwieldy and prohibitively expensive. The present capability to produce ultra-reliable systems is a result of the general availability of fairly high-reliability components. To realize this capability fully, it is necessary to obtain the optimum benefits afforded by the use of redundancy.

The various ways in which redundancy can be employed to improve overall system reliability are reflected by different configurations of nonredundant system components. In order to compare different configurations it is necessary to determine the reliability of the overall system. System reliability, which is the probability of system success as a function of time, can be determined in two ways. The first is to gather statistical data about the failure history of a number of sample systems and use statistical analysis to find the reliability function for other systems of the same type. The second is to predict a total system reliability on the basis of known component reliabilities and the configuration of the component interconnections. In practice, neither method yields more than a good approximation to the actual reliability function.

The validity of the first approach, statistical failure analysis, depends upon the number of samples used and the length of time over which they are observed. The more samples and the longer they are observed, the more accurate the reliability prediction will be. For this reason, statistical analysis based upon gathered statistical data is usually restricted to relatively small components, such as integrated circuits. There is generally neither time nor sufficient samples to obtain meaningful data for a large complex system.

The second approach is to predict system reliability on the basis of known component reliabilities and the configuration of these components. This is more pragmatic than, but not potentially as accurate as, statistical analysis of the whole system. Prediction must rely upon assumed reliabilities of component interconnections as well as the reliabilities of the components themselves. In addition, the reliability of software and other system algorithms should be taken into consideration, but until recently,[10] little had been done in this area. However, configurational analysis is the method most often used to predict system reliability, because it produces results faster and at lower cost than statistical analysis of the whole system.

A nonredundant system, which together with other nonredundant systems constitutes a redundant system, will be called a simplex system, a simplex element, or just an element. If it is assumed that the interconnections among simplex elements can be associated with specific elements so that the interconnections need not be reckoned with explicitly, then configurational analysis will, in theory, provide a reliability function for the redundant system that is as accurate as the reliability functions for the simplex elements. Some of the more relevant work in this area has been done by Carter[2] and Mathur.[7] However, these analyses assume statistical independence of the simplex elements and ignore the problem of systematic failures. That is, they fail to consider the effects of statistical correlations among the simplex elements of the redundant system. (Klaschka[6] does discuss statistical interdependence, but it is among serial components of the system, not among redundant elements.)

Systematic failure modes have a detrimental effect on reliability in that they diminish the benefits of redundancy, that is, the actual reliability of a redundant system is not as high as conventional configurational analyses would indicate. A simple example demonstrates that the effect of systematic failures is non-negligible. Consider a computer configuration where three computers run in synchronization and their outputs are voted upon. (The Saturn V guidance computer is such a configuration.) Practical considerations dictate that software be considered in an analysis of the system reliability, for, after all, if the software does not work properly then the system will not work properly. Since the three computers all run the same program simultaneously, it is clear that an error caused by faulty software totally defeats the redundancy. Apollo experience shows that a sizable percentage of system malfunctions stem from software and crew procedures, so for configurations like the one described, systematic failures are a first order effect. It will be seen that systematic failure modes are a first order effect for other redundant configurations as well.

The notion of systematic failure can be quantified by considering the statistical correlations among the simplex elements of a redundant system. Perhaps the reason that current approaches to redundancy analysis do not consider systematic failure modes is that the analyses use information only about the operational configuration of the system. The configurational information, and the simplex reliabilities, by themselves, are not sufficient to determine the correlations among the simplex elements. More information is needed. It appears that some of this information can be extracted from a timeline of the system development.

This study attempts to do two things. The first is to develop an approach for determining correlations from an analysis of the development timeline of a given system. The second is to include these correlation factors in the equation for overall system reliability.

The system development timeline consists of various activities, such as specification, design, and production. The interaction of these activities can be represented by a system development structure. In order to make use of this structure it is necessary to understand how erroneous behavior propagates through the operational system and how that erroneous behavior can be related to the system development structure.

The model that is developed considers all erroneous behavior to be related to flaws which have been built into the system. If such a flaw is activated during system operation, then erroneous behavior will result. A flaw in a development activity can be carried along through other development activities in the development structure and eventually be built into the system. If activated, it can propagate through the operational system and eventually cause the system to malfunction. By analyzing the paths that a flaw can take through the system development structure, and the possible paths of erroneous behavior through the operational system, it is possible to determine correlations among the simplex elements of the redundant system.

Chapter 2 develops a model for erroneous behavior. The discussion is concerned with the relationships between external events and the erroneous behaviors that they trigger.

Chapter 3 defines the activities that make up the system development structure, then analyzes the relationships among the activities of a structure for a simplex system, and the ways in which flaws are inserted and carried through the structure. Finally, a structure for a redundant system is presented, and the means for determining correlations are discussed.

Chapter 4 is concerned with the mathematics of correlation. It develops a number of different approaches for including correlations in the reliability equation for the system. One of these approaches provides a straightforward method for accurately including the effects of correlations in the analysis of complex redundancy configurations. Finally, it is shown that even very small correlations can have a detrimental effect on overall system performance.

Chapter 5 draws conclusions from the results of the study.

CHAPTER 2

A MODEL FOR ERRONEOUS BEHAVIOR

2.1 Motivation

'Systematic failure' implies related erroneous behavior among corresponding simplex elements of a redundant system. In order to study the effects that systematic failure modes have on the reliability analysis of a redundant system, it is necessary to develop a model for erroneous behavior. The model should satisfy the following requirements:

1. It should be as general as possible and avoid dependence on any particular system configuration.
2. It should accommodate the characteristics of different modes of erroneous behavior.
3. It should make use of information available from the development timeline of the system.
4. It should provide a precise terminology so that ideas can be presented and clearly understood.
5. It should be self-consistent.

The basic assumption of the model is that erroneous behavior stems from flaws that have been built into a system. Erroneous behavior results when a flaw is activated by some influence that is external to the system. In order to pursue the ramifications of this assumption it is necessary to define and discuss the concepts which will become part of the model.

The notion of system is one of the more basic concepts. A system is a bounded collection of hardware and algorithms. In principle, the boundary can be arbitrarily chosen to include any subset of a larger collection of hardware and algorithms. In practice, such a larger collection exhibits natural subdivisions. For example, a spacecraft is a system. It can also

be subdivided into a set of systems. One such set might be a superstructure system, a propulsion system, and an avionics system. (In the technical jargon these are often referred to as subsystems—for the obvious reasons.) Another subdivision might combine the left half of the spacecraft superstructure with the inertial measurement unit, and combine the right half of the superstructure with the computer memory. If the boundaries are definitively drawn, this is a valid subdivision—everything inside each boundary is a system. However, the nature of this latter example does not make it particularly useful.

Implicit in the concept of system is the notion of hierarchical level. On an absolute scale, the top-level system is the universe. The universe can be divided into subsystems, and these into sub-subsystems, and so on. From an engineering viewpoint the top-level system includes the total collection of hardware and algorithms which are part of a given project. For example, in a space transportation system such as the NASA space shuttle, the top-level system might include the launch site(KSC), mission control(MSC), and the vehicle itself(SSV). This collection (KSC + MSC + SSV) can be subdivided as many times as is meaningful to a particular analysis.

The next step is to determine what happens when a system acts in an erroneous manner.

2.2 Modes of Erroneous Behavior

The phrase 'erroneous behavior' is used axiomatically to imply the occurrence of an event which is neither expected nor desired. Erroneous behavior that is confined within a system boundary is not visible from outside the boundary, and therefore cannot affect anything outside of the boundary. However, if erroneous behavior propagates from the inside to the outside of a system boundary, then, by definition, it results in a system malfunction. A system malfunction occurs when an event or ordering of events in a definitively bounded system results in erroneous behavior crossing the

system boundary and passing out of the system. The occurrence of a system malfunction always implies that erroneous behavior has occurred in the system, but the converse is not true.

There are three modes of erroneous behavior; they are errors, failures and faults.

1. An error is an incorrect numerical or logical value resulting rom an observation or calculation. Errors result from algorithmic flaws in the system, erroneous inputs to the system, faults, and other errors. Errors can be divided into two types on the basis of immediate cause. An error of the first type is the immediate result of a fault or an activated flaw. An error of the second type is the result of another error or a certain class of erroneous inputs to the system. A chain of propagating errors can contain at most one error of the first type.

2. A failure is an alteration of hardware that causes loss of physical integrity. A hard failure means a permanent and continuous loss of integrity. A transient failure means a sporadic loss of integrity (i.e., specific hardware characteristics), such as an intermittent diode. It is possible for a failure to cause another failure directly. A failure that entails high voltage arcing, or an explosion, for example, will probably cause other hardware to fail.

3. A fault is an incorrect representation of a value due to a failure or to some external physical influence such as noise. The best example of a fault is an incorrect logic level. This is different from an error, and does not result in an error unless and until it is observed.

The modes of erroneous behavior are related in a straightforward manner to the abstract and concrete aspects of the system. An error is

erroneous behavior of an algorithm; a failure is erroneous behavior of hardware. A fault is the link between the two. An error propagates when the erroneous output of one algorithm serves as the input to another algorithm. Failures can also propagate, but a situation in which a failure directly causes another failure is usually catastrophic, such as an explosion or a fire. A far more common occurrence is that a failure results in a fault, which, in turn, results in an error. A fault, unlike failures and errors, cannot propagate erroneous behavior of the same mode. That is, a fault cannot cause another fault, since, by definition, the 'observation' of the first fault results in an error. Either a fault produces an error or it is never observed and has no effect.

Erroneous behavior either propagates in a system until it crosses the system boundary to produce a system malfunction, or else it dies out. Such a chain of erroneous behavior must be triggered by something external to the system. In the case of the top-level system, i.e., all the hardware and algorithms under consideration, these external events come from the environment. In the case of a subsystem, the external events may come from the environment or may be the result of a system malfunction of another subsystem.

2.3 Events which Trigger Erroneous Behavior

According to the basic assumption of the model, erroneous behavior results when a flaw which has been built into the system is activated by an external event. This means that certain external events can cause erroneous behavior because of the existence of a system flaw. A flaw is a weakness in a system which allows external influences to prevent the system from meeting its intended goals. A flaw can get into the system during any of the various activities which occur during the development of the system. These activities and their interrelationships are the subject of Chapter 3.

External events (external influences) fall into two categories. The first is comprised of the natural external events: physical stress, noise,

and erroneous inputs to the system. The second is comprised of the quasi-external events: coincidental circumstances and normal operation of the system. Although the quasi-external events are defined to lend consistency to the model, they are not as artificial as they first appear.

The quasi-external events are both related to time. Coincidence, or coincidental circumstance, is the random timing relationship among parallel processes in asynchronous parts of a system. Certain timing relationships between different parts of the system may result in erroneous behavior. For example, consider the case of an undetected race condition in digital circuitry. The existence of the race condition is a flaw in the circuit design, but it does not impede the proper operation of the circuit as long as the phase relationships among inputs and outputs are within certain bounds. If the circuit inputs are asynchronous, however, the phase relation of the inputs cannot be guaranteed to be within the necessary bounds. When the phase relation of inputs exceeds these bounds, it is said to be due to coincidence. Another example is incompatible software running on multiple collaborating processors. The incompatibility is only detrimental to the system if the timing relationships in the software happen to be unfavorable and cause the incompatibility to be realized. The flaws are activated by coincidence in these examples because the timing relationships that cause parallel processes to interact erroneously are stochastic.

There is a certain class of algorithmic flaw which does not appear to require any activation at all. The characteristic of these flaws is that they produce errors by mutilating good data rather than by accepting bad data. An example is a software routine that loses data precision by improperly rounding intermediate data. The more blatant flaws of this type, such as incorrect program branches, are usually caught early in the system testing, so it is unlikely, though not impossible, for them to exist in the finished system. Even when such a flaw does get into a system, however, it cannot produce errors without being activated. The activation occurs when the system, in progressing through its normal operation,

invokes the flawed algorithm. Normal operation is the quasi-external event which triggers the type of flawed algorithm that produces an error by unilaterally mutilating good data.

The natural external influences result from the environment or the malfunction of another system. Those which affect hardware are physical stresses and noise. Physical stress is that which causes loss of integrity of hardware (failure) in conjunction with a system flaw. If the failure is caused by a stress greater than the maximum specified level, then the specification is not correct. If the failure is caused by a stress less than the maximum specified level, then there is a flaw in the design or production of the system.

Noise is a disturbance of the characteristics of an information carrying medium. This disturbance creates a change in information without altering the physical integrity of hardware. A good example is electromagnetic noise which alters the value of bits sent over a transmission line. But noise also applies to information carried by hydraulic, optical, or mechanical methods. For example, brushes on a shaft angle encoder could be shaken sufficiently to cause an incorrect readout without actually damaging the hardware.

Noise causes faults in conjunction with a system flaw. If a fault is caused by noise which is greater than the maximum level specified for the system, then the specification is not correct. If the fault is caused by noise less than the maximum specified level, then there is a flaw in the design or the production of the system.

The natural external events which cause errors directly are erroneous inputs. An erroneous input occurs when incorrect data or commands that originate outside of the system boundary enter the system. Erroneous inputs are of two types: those which are recognizably erroneous and those which are unrecognizably erroneous.

An input which is recognizably erroneous will be detected by a flaw-free system. When such an input is detected, corrective action can be taken, so that the input does not result in an error. However, a system flaw may prevent the detection of such an input, in which case an error occurs. By way of example, consider the situation where certain program options are selected by inputting a number to the computer. The number is used as an index to a program transfer table, so an input greater than the range of the table is erroneous. Proper testing of the input before the transfer is attempted will detect all out-of-range inputs before they can do damage. Failure to perform such a test allows the erroneous input to cause an error. Such a situation occurred in Apollo when an astronaut inadvertently entered star #0 into the computer during a navigation operation, causing an error that resulted in a computer restart.

In general, if the potential values for an input can be classified as valid or invalid, failure to test for invalid values produces a flaw in the system. Recognizably erroneous inputs must act in conjunction with such a flaw in order to produce an error. An error so produced is of the first type because it results directly from a flaw.

An input which is unrecognizably erroneous cannot be detected by a flaw-free system. Such an input is a member of the set of valid values, but an inappropriate member of that set with respect to the prevailing circumstances. A simple example is that of a sensor input to a computer. The computer may be aware of an acceptable range for the sensed value, but there is no way it can tell if an input within that range is, in fact, the proper value.

An error caused by an unrecognizably erroneous input is not the direct result of a flaw, so such an error is of the second type. The fact that an unrecognizably erroneous input can induce an error without acting in conjunction with a system flaw is an apparent exception to the model under discussion. The exception exists at a subsystem level, where a system

malfunction of one subsystem can generate an erroneous input to another subsystem. A top-level system that interacts only with a natural environment does not exhibit this exception, however, because inputs from a natural environment cannot be erroneous.

The relationships among the modes of erroneous behavior and the external events that activate them are shown in Fig. 2.1.

2.4 Relationship of Flaws and External Events to the Reliability Function

The model under discussion relates erroneous system behavior to built-in system flaws that are activated by external events. For the model to be reasonable, it must be possible to include the notions of flaws and external events in the reliability function. Any existing reliability function should be expressible in a form which includes variables that represent both flaws and external events.

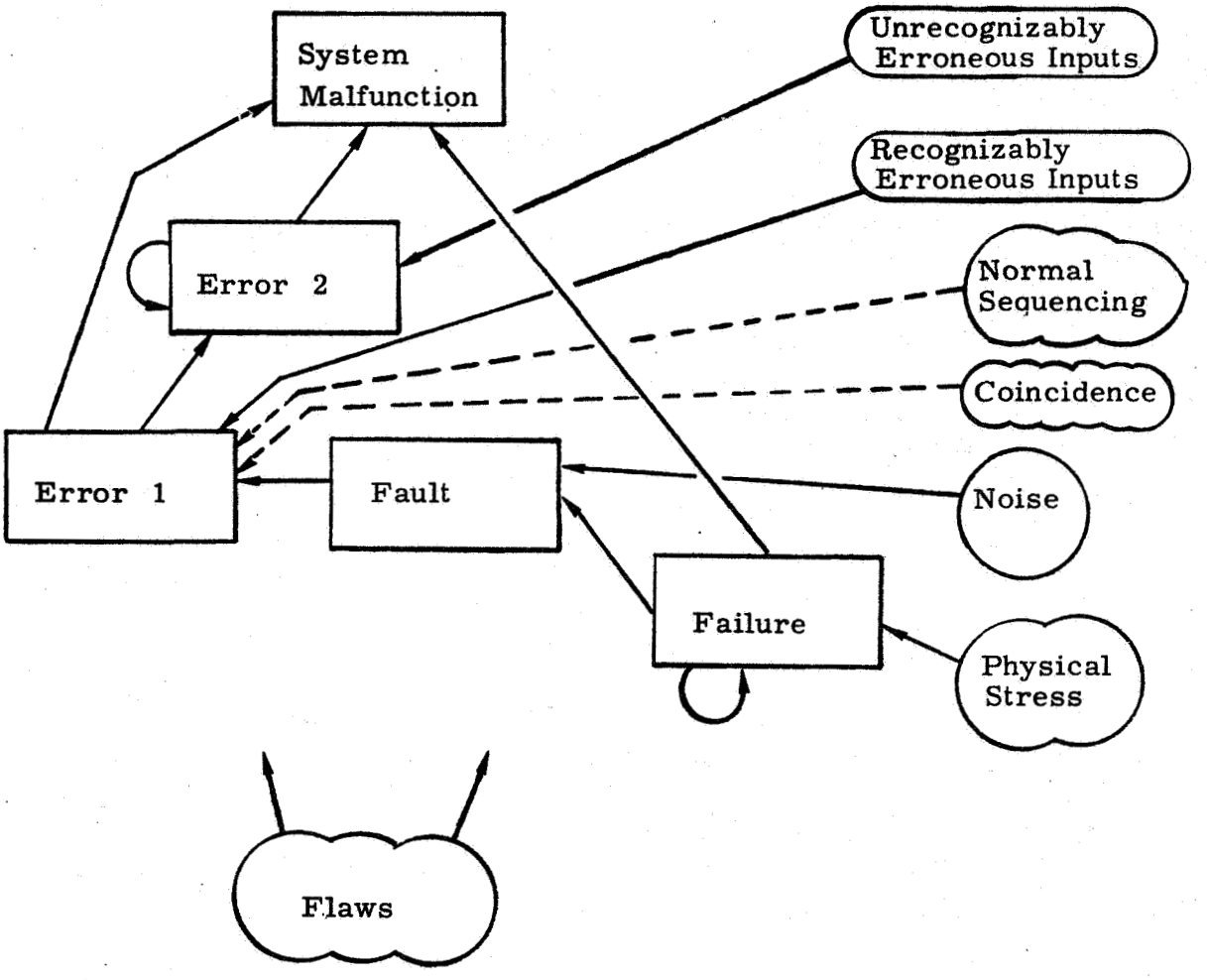
Once a system is completed the number of flaws that it contains is fixed at a constant value. This value, or a constant function of this value, will be called the flaw density of the system (denoted by the letter D).

The occurrence of external events is expressed as a rate variable (denoted by $\lambda'(t)$). The rate of occurrence of external events may be a function of time. In a spacecraft, for example, $\lambda'(t)$ would probably change as the vehicle moves from the earth environment to a space environment.

From Shooman,[14] the general reliability function can be expressed as

$$R(t) = \exp\left(-\int_0^t z(\xi)d\xi\right) \qquad (2.1)$$

where z(t) is the hazard rate for the system. This hazard rate is analogous to the rate of occurrence of external events. The form of (2.1) will be maintained if $z(t) = D\lambda'(t)$, and since D is constant, z(t) can always be



Erroneous Behavior Modes and the Events that Activate Them

Fig. 2.1

decomposed in this manner. This means that for any reliability model, R(t) can be expressed in terms of D and $\lambda'(t)$.

Consider the reliability model in which failures occur at random. For this model the hazard function z(t) is a constant equal to λ, the failure rate. The reliability function is obtained from (2.1).

$$R(t) = \exp(-\lambda t) = \exp(-D\lambda' t) \qquad (2.2)$$

So for this case, the product of the flaw density, D, and the rate of occurrence of external events, λ', is equal to the system failure rate, λ. The response of the system reliability to D and λ' is as one would expect. In the absence of flaws, the reliability equals one; and an increase in the number of flaws, or the in rate at which they are triggered, causes a decrease in reliability.

The erroneous behavior model that has been presented states that a flaw is a contributing factor in each case of a system malfunction. This serves as a basis for analyzing systematic failure modes by examining correlated flaws which occur in different simplex elements of a redundant system.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 Motivation

The purpose of studying the development timeline of a system is to determine the ways in which flaws become incorporated into the system. In a redundant system this knowledge can be used to analyze the relations among flaws in the simplex elements. These relations indicate possible statistical correlations.

While the existence of correlations can be extracted from the structure of the system development, the degree of these correlations cannot be determined exactly. In some cases the degree of correlation (the value of a correlation coefficient) can be determined a priori on the basis of intuition and engineering judgement. For example, the Saturn V guidance computer mentioned in Chapter 1 is a TMR system that exhibits complete correlation among the three "copies" of software. This is a straightforward conclusion since the three copies of software are identical in form and are executed in synchronism. In most cases, however, the degree of correlation is not so obvious.

Engineering judgment can uncover certain trends in correlations, and perhaps even determine the relative degrees of different correlations, but in general, cannot determine the absolute degree of a correlation. This problem can be attacked by employing information about previous systems with similar correlations. A simple estimate of the value of a correlation coefficient is the mean of the correlation coefficients observed for similar correlations in previous systems. Before correlations can be quantitatively determined, however, their potential existence must be uncovered.

3.2 System Development Structure

System development can be broken down into a number of stages, each of which can be divided into different activities. These activities form a two-dimensional hierarchy where the vertical dimension represents the sequence of stages from system conceptualization to operation, and the horizontal dimension depicts the distinctions among physical hardware and the various levels of algorithm.

3.2.1 Vertical Dimension

The first stage of system development is to define the problem. The Problem statement is a description of the objectives to be met, or the goals to be achieved. It may seem hard to conceive of a flaw occuring in such an apparently straightforward step. If, for example, the desire is to land a man on the moon, then this objective is so stated and that is that. However major projects of this sort usually have a host of minor objectives, some of which may be stated so briefly or ambiguously as to cause misinterpretation by those who implement the system. Minor objectives may be incompatible with one another. If different teams implement different objectives that are in subtle conflict, a resulting flaw may not be found until a system malfunction occurs.

The Environment model is a description of the physical laws, characteristics, and constraints of the environment and materials associated with the problem. This information serves, along with the problem statement, as an input to the specification process. A flaw in the environmental model may be something as blatant as using the wrong equation of motion in a particular situation or something as subtle as failure to include higher order perturbations in an equation of motion.

The specification process determines what the system must do and what constraints it must meet in order to accomplish the stated objectives. Specifications are certain requirements that the system must fulfill.

Examples of somerequirements for a computer oriented system are limits on physical size, weight, and power consumption, and a minimum computational throughput. A specification flaw is the result of these requirements being stated incorrectly or incompletely.

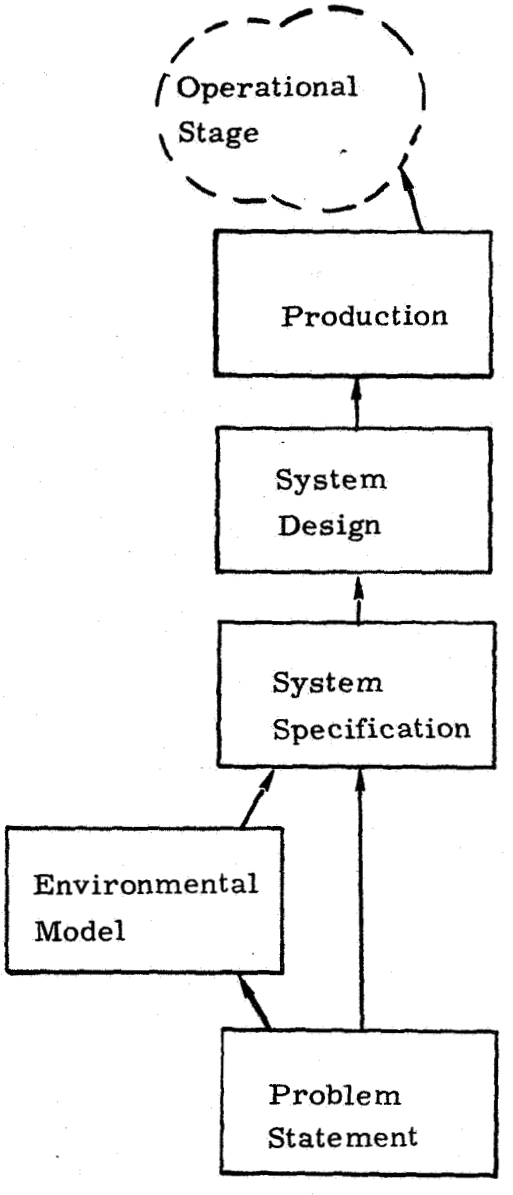
The system design process determines how the specification is to be implemented. This is, in effect, a "second level" specification to whomever must realize the system. It is in a form that can be translated into physical hardware and into the algorithms that will reside on the hardware. A design flaw results when the design does not properly implement the requirements of the specification.

Production is the realization of the system design, a commitment of the system to a final form. The form of realization depends upon the particular system aspect under consideration. (The various system aspects are discussed in Sec. 3.2.2). The production of hardware is clearly different from the production of software. Hardware production is a physical realization of the hardware design. Software has no physical realization, per se, but there are by-products such as assembly listings and tape or card representations which can be used to load hardware. A production flaw results when the realization of the system does not properly implement the design.

The stages of system development are shown in Fig. 3.1. Each of these stages could be decomposed further, but such a level of detail is not necessary for the purposes of this analysis.

3.2.2 Horizontal Dimension

The various development stages can be expanded horizontally by considering the physical and the algorithmic aspects of the system. The physical aspects of the system imply hardware. The algorithmic aspects imply hardware function, firmware, software, and procedure.



Vertical Dimension of the Development Structure

Fig. 3.1

An algorithm is a structuring of the physical hardware or a logical state imparted to the physical hardware such that it behaves in a desired manner. Algorithms can be categorized into a hierarchy, the elements of which are keyed to, but not restricted to, computer systems. The higher level algorithms are those with the greatest scope—they are general in that they are related to the system as a whole. The lower level algorithms are more restricted in scope—they are related to specific parts of the system.

The most rudimentary algorithms are those related directly to the physical hardware. Hardware function describes those algorithms which are intimately associated with the physical structure of the hardware. One example of this is digital logic which composes the control registers and timing generation of a CPU. Simple combinations of Boolean operations such as adders and shift registers also fall into the realm of hardware function. A less obvious case is the structure of mechanical linkages in a hydraulic actuator.

The next higher level of algorithm is concerned with the state of the hardware, but not the physical structure. Firmware describes algorithms that operate from a stored program which is normally a fixed design parameter and does not vary from application to application. State control algorithms in a computer are firmware. The name given to state control algorithms depends upon their particular implementation. RAM and ROM implementations are usually called microprogram. Gate-level implementations are referred to simply as control logic. Although the latter implementation may be mistakenly cconsidered hardware function because it has no explicit program structure, it is nonetheless firmware, for it realizes the same algorithms as the equivalent microprogram implementation.

Software implies stored program algorithms which normally vary from application to application.

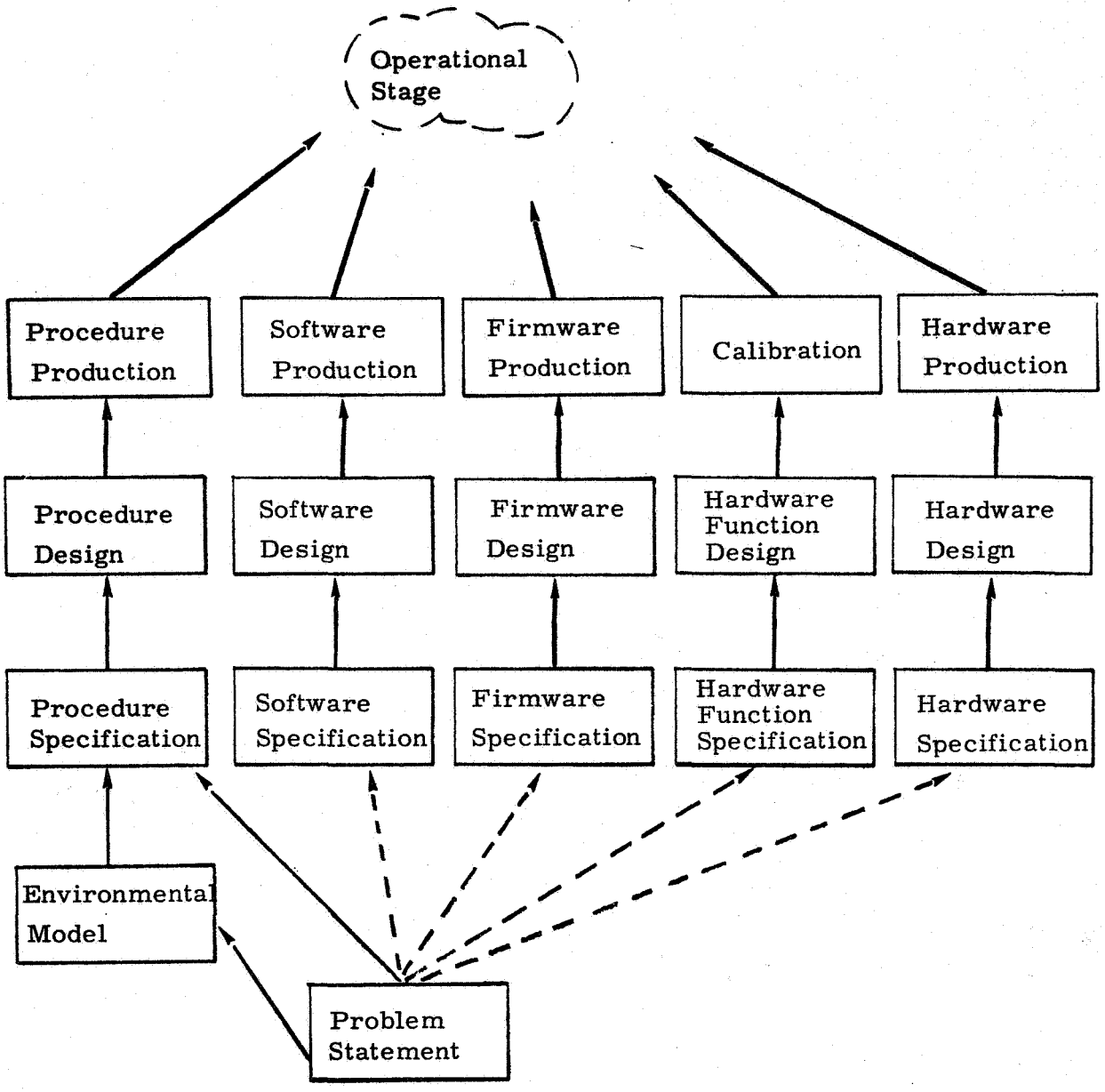
Procedure refers to the most general category of algorithms: the major steps taken to execute the operation of a system in such a way as to achieve its intended objectives. Man is usually a participant in procedural level operations. An example of procedure is the steps taken prior to and during the landing of an aircraft.

Physical hardware is the vehicle in which the algorithms reside. The algorithms, in turn, impart a "personality" to the hardware. These two aspects of the system must co-exist, for neither is meaningful alone.

The characteristics of the specification and the design stages of hardware development and algorithm development are about the same. However, the characteristics of hardware production and algorithm production differ noticably. Algorithm implies an abstract logical structure, so most of the "work" has been done during the design stage. Production here implies some physical representation of the encoded algorithm structure. Procedure, for example, is usually printed in a manual, but it could be committed to microfilm, or to a digital representation in memory that would be used as input to a display system. Software production usually results in a tape or some representation which can be used to load or produce a memory state in a computer. A by-product of this is an assembly or compilation listing. Firmware production is roughly the same as software. Hardware production means a physical realization of the design and the "loading" of algorithms into the hardware. Factors include procurement and assembly of parts, and controls of these operations.

Hardware function is only produced in the sense that the building of the physical hardware reflects the hardware function. However, there is a production activity which is closely allied with hardware function. This is calibration—the final adjustment, prior to operation, of variables closely associated with the hardware function.

Both dimensions of the development structure are shown in Fig. 3.2.



Horizontal Expansion of the Development Structure

Fig. 3.2

3.2.3 Hardware versus Algorithm

The distinction between hardware and algorithm is an important one. Algorithmic flaws and hardware flaws result in different modes of erroneous behavior: hardware flaws are responsible for failures, algorithmic flaws are responsible for errors. Correlations are determined by the ways in which potential flaws relate to erroneous behavior modes, so it is advantageous to separate the hardware and algorithmic aspects of a system as completely as possible.

Procedure, software, and firmware are clearly different from hardware. The major conceptual difficulty in distinguishing hardware from algorithm is in the separation of physical hardware and hardware function. The distinction is more obvious for digital systems than for others, but in general, it can be considered to be the difference between the physical and the abstract. Digital logic performs a specific function. This function is clearly different from the components of which the logic is composed. The functional logic design can be done independent of the layout and production of the hardware components. The algorithm-hardware distinction can be more subtle in other cases. The following example considers a simple system component which demonstrates some of the conceptual difficulties that can be encountered.

A thermostat is a threshold device which closes a circuit when a predetermined temperature is reached. This behavior is a function. The hardware is the bi-metallic strip that is connected to the contact to be closed. The notion of function can be extended to the bi-metallic strip, for this has a certain function of temperature designed into it. Going one level deeper, each of the two different types of metals has a function associated with it: the coefficient of thermal expansion. This function is so integrally associated with the hardware that it is almost (but not quite) stretching a point to make a distinction. Whoever is responsible for procuring the hardware to build a thermostat could simply purchase a ready-made bi-metallic strip with the appropriate properties and let it go at that, much

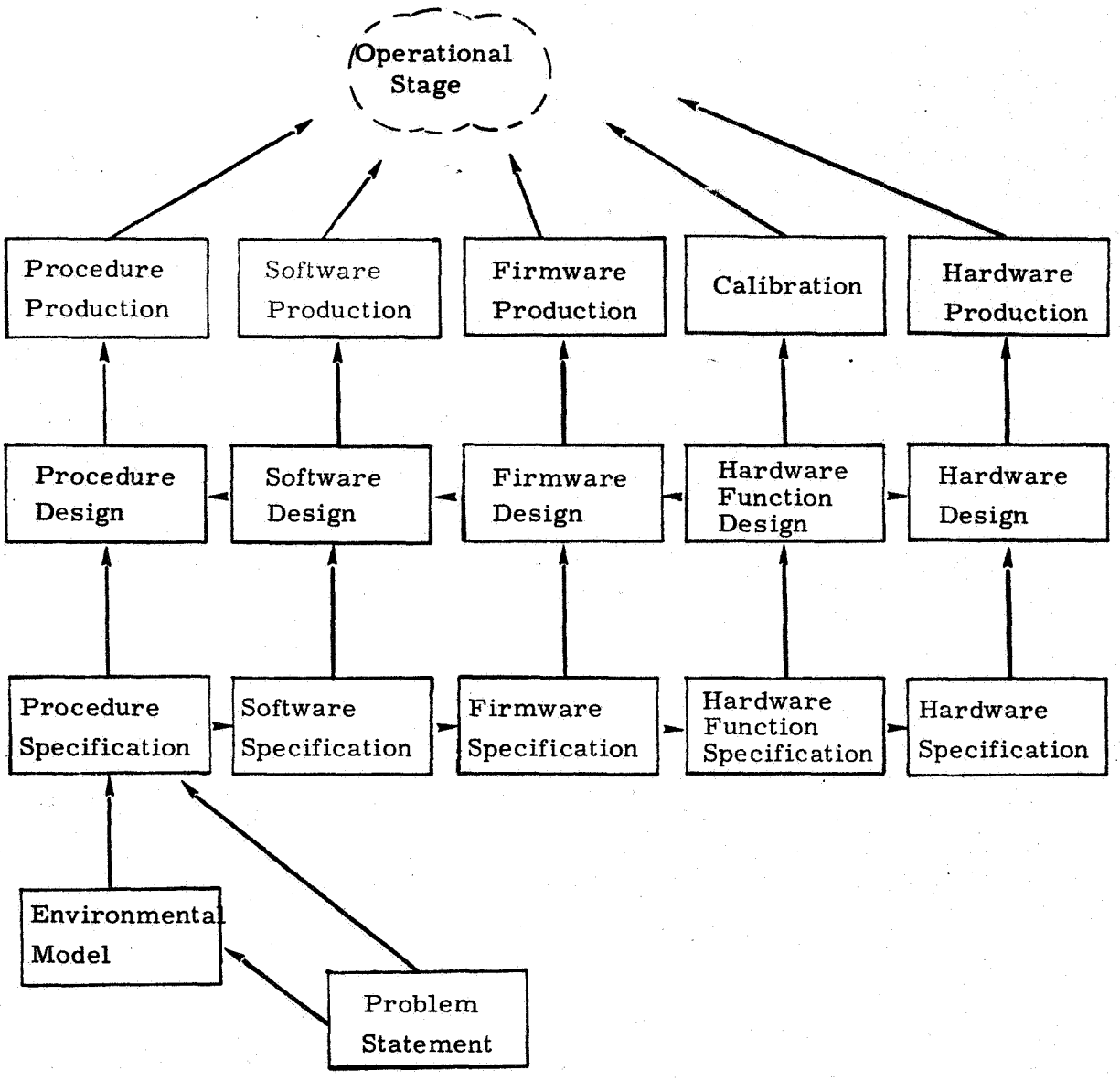
the same way as one buys an integrated circuit. But the fact that his level of interest does not extend beyond the higher level function does not mean that the lower level function is not there. The person responsible for metallurgy must worry about the purity of each metal, or the accuracy of the alloy, in order that it provide the proper coefficient of expansion.

The level to which it is necessary to go in differentiating hardware from hardware function depends upon the level of the system under consideration. An oven manufacturer, for example, is interested only in the temperature at which the thermostat will close. The thermostat manufacturer, on the other hand, is keenly interested in the coefficient of expansion of the alloys he is using.

3.3 Interaction of Development Activities

The system development structure shown in Fig. 3.2 can be further refined to include other interactions among the various development activities. An interaction, the influence of one activity on another, implies that the output of the one activity serves as an input to the other. These interactions are not arbitrarily chosen; rather, they reflect influences that actually occur during the development of a system. Fig. 3.3 shows a general system development structure which can be adapted, with minimal effort, to most digital systems.

If the structure is viewed as a game-plan for system development, then the particular choice of interations shown in Fig. 3.3 reflects a top-down development philosophy—an approach to system development which is analogous to the structured programming of Dijkstra.[3,4] Top-down development provides excellent visibility into the system development effort and provides maximum consistency among the activities associated with that effort. Because there is no feedback in the structure, an activity cannot affect a predecessor activity, so conflicts are eliminated. Any change in an activity must regenerate all succeeding activities in order to maintain overall consistency. In practice, systems are not designed this way: there



General Development Structure

Fig. 3.3

are feedback loops. Fig. 3.3 could be modified to reflect these feedback interactions, but the resulting picture would be greatly complicated, and would defeat the purpose of the top-down development structure.

When the development structure is used as a tool for analyzing flaws and flaw propagation, it is not necessary to include feedback interactions that reflect the actual course of the system development, because the aim is to model the final sequences of flaws that result in correlations. The development structure serves to relate flaws to possible occurrences of erroneous behavior. This relationship is represented by a chain of influences from a flawed activity, through other activities, to a possible instance of erroneous behavior. The relationship is established by the fact that a flaw is actually transmitted forward from activity to activity in the chain. It is irrelevant that the flaw may have iterated several times before the transmission actually took place.

The development structure shown in Fig. 3.3 exhibits two kinds of influences: those in the vertical direction and those in the horizontal direction. The vertical influences are relatively straightforward—each stage of system development lays the foundation for the next stage. The horizontal influences relate the various system aspects during each stage of system development.

The horizontal influences at the specification stage are directed from procedure specification to physical hardware specification. The process of specification means determining what the system must do, so the determination of the more general aspects of the system will influence the determination of the particular aspects.

Procedure, the highest level algorithm is, of course, the most general. Once procedure has been determined, it is used as a constraint in determining what software, the next level algorithm, must do in order to support the procedure. For example, in the Apollo project, procedural specification

determined the various phases of the mission and what was to take place during each one. The choice of mission and mission phases created requirements for what the software would have to do in order for the system to execute these mission phases.

A relation similar to the above holds between software and firmware. Once the job of software has been determined, it serves as an input to the process of determining the nature of the instruction set that will best support that software.

The choice of an instruction set influences what the computer logic (hardware function) must be able to do in order to realize that instruction set.

The last relationship at the specification stage is that the hardware function specification influences the physical hardware specification. The physical hardware must be able to support the desired architecture. In actual practice, the state-of-the-art may not be developed to the point where there is a hardware technology capable of doing this, because of size, weight, power, and speed limitations. This potential conflict creates problems in a top-down design effort, for there has to be a feedback loop from hardware specification to procedure specification, in order that consistency of specifications be maintained. From the flaw analysis point of view, however, the loop can be ignored, since the object is to detect flaws, not to correct them. If the inconsistency is not resolved early in the system development, and the system is built around an inconsistent specification, then the flaw which is associated with this inconsistency is considered to come from the procedure specification. The procedure specification may be said to be unrealistic because it cannot be implemented within the state-of-the-art.

The horizontal influences at the design stage are not unidirectional like the influences at the specification stage. The pivotal point of influence is the hardware function design activity. For digital systems, this activity

generally entails the design of digital ligic, but there are also other considerations such as environmental control for the system, accessibility to the operators, and packaging interfaces with other systems.

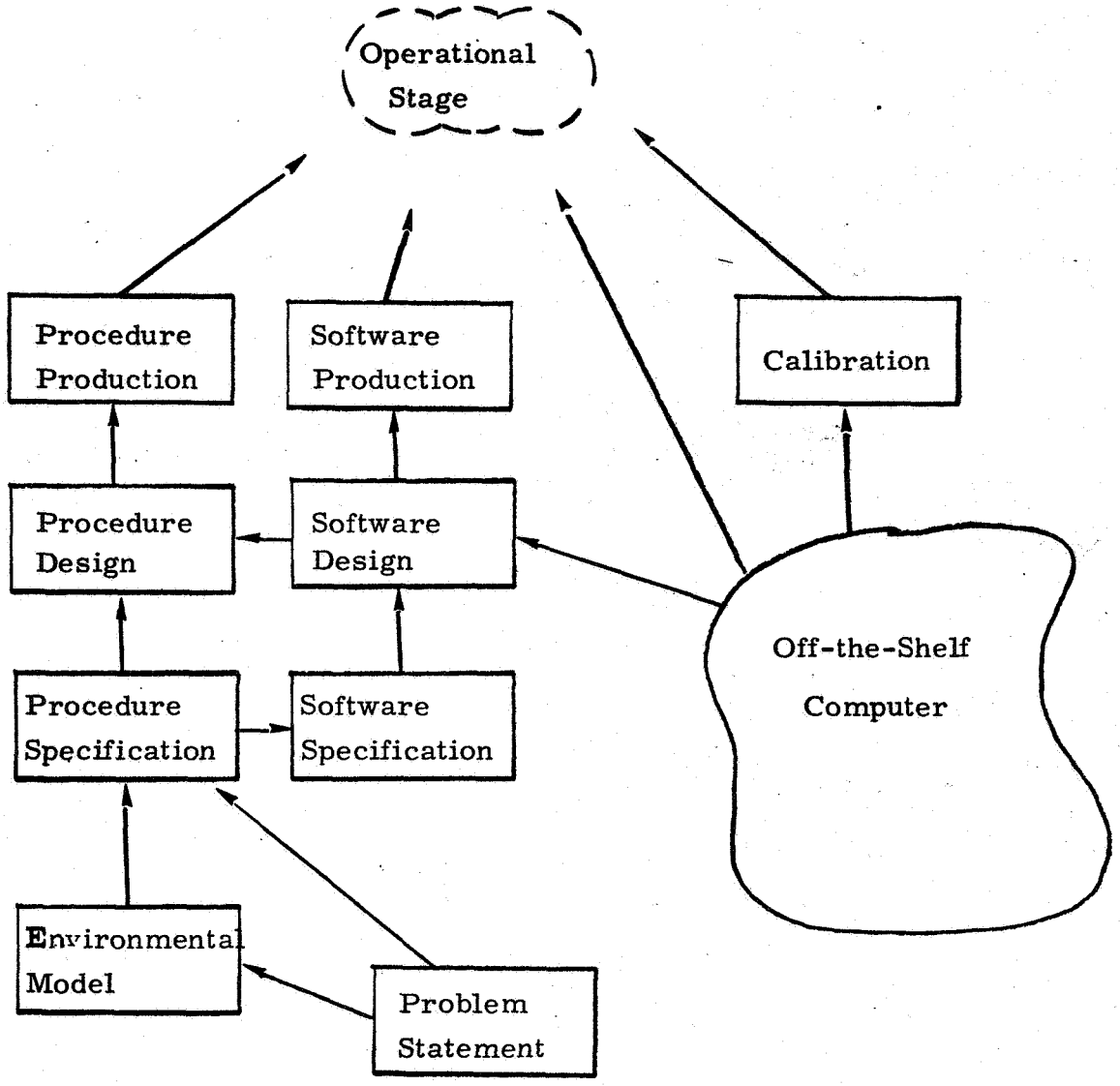
Hardware functional design directs physical hardware design because form follows function. Clearly, hardware must be organized to support the desired function—the wings of an airplane must be shaped to provide the necessary lift—digital logic must be interconnected so as to properly model the desired boolean equations.

Hardware functional design also directs the design of the higher level algorithms. Only when the characteristics of the digital logic have been clearly defined can a microprogram be written. The design of the microprogram fixes the details of the instructions so that the software can be clearly defined. The detailed operating procedure, that is, which buttons are to be pushed in response to what, is determined by the specifics of the software.

There are no horizontal influences at the production stage. Production activities for the hardware and the various levels of algorithm are independent of one another.

There are other interactions which could take place in the development structure. For example, if firmware specification were to delineate the fine details of the instruction set, then this activity could affect the software design directly. However, such additional influences are not necessary for the current analysis because a path which covers this case already exists. Any accuracy that additional influences add to the model is overshadowed by the fact that they complicate the analysis.

Modifications to the structure in Fig. 3.3 are necessary when certain activities are consolidated. For example, if a development constraint is to use an off-the-shelf computer, consolidation takes place as in Fig. 3.4.



Effects of a Development Constraint (OTS Computer)

Fig. 3.4

Each activity can be represented by a more refined substructure. In particular, a consolidation, as above, can be expanded to reflect the development effort of the computer. But unless information can be gained by such an expansion, it is best to keep the structure as simple as possible.

3.4 Flaw Insertion and Propagation

Every flaw in a system can be associated with a particular system development activity. The manner of flaw injection during an activity can be categorized as follows:

1. Misinterpretation of the output of the predecessor activities. This involves such things as a designer misreading a specification, or assuming that it means one thing when it really means another.

2. Erroneous translation of the predecessor activities. For example, the person who translates a specification into a design may fully understand the specification, but may err for some reason and produce a design which does not reflect that specification.

3. Erroneous transcription of the results of an activity. These are generally mistakes in documentation, confusing or ambiguous documentation, and typographical errors.

4. Erroneous transmission of information to the next activity. This entails interface problems such as sending the wrong update of a design and papers getting lost, mixed up, or even being altered in transit.

For a flaw in any activity to propagate, it must be carried along by the development effort in order to exist in the final system. For example, a software specification flaw can be reflected in the software design which in turn is reflected in the software production. It is necessary for a flaw to be reflected in some production activity or else it cannot be considered to exist in the finished system. This is seen as a chain of influences by one activity on another, starting with the activity which generated the flaw and terminating with some production activity.

A flaw that has propagated to another activity does not change its basic nature. If the new activity is consistent with the flawed activity which preceded it, then it reflects the flaw. For example, a software specification flaw may be reflected in the software design, but it does not become a software design flaw.

The fact that a flaw can propagate does not mean that it will. Propagation depends upon the characteristics of the influence that a flaw has on activities that it enters, and upon the possibility of compensating flaws cancelling each other.

Influences of one activity on another have two characteristics: strength and directness. Strength means how much a receiving activity depends upon the information sent from an influencing activity. Directness is an indication of the amount of interpretation done by the receiving activity.

The influence of design on production is generally very strong and direct. There is little room for interpretation of design information—the translation of this information is very mechanical in nature. For example, a logically incorrect program, the result of a flaw in the software design activity, will almost surely contaminate the output of the software production activity. A compensating flaw in the production of a program load tape is highly unlikely.

The influence of specification on design is also strong, but not as direct as the influence of design on production. The translation from specification to design leaves room for interpretation. Consider a flaw such as an incomplete specification. The software design could reflect this flaw, for example, by failing to test the ranges of certain input variables. But the design activity might put in the tests, even though the are not called for in the specification, and this would cancel the effects of the faulty specification.

Most influences in the horizontal direction are less strong and direct than those in the vertical direction, and therefore weaker with respect to propagating flaws. Consider a poorly designed program which may request operator action at an inopportune time. This software design flaw would most likely be reflected in the procedure design. On the other hand, many software design flaws, like loss of precision in a complex computation, have no effect on the design of operating procedure. For another example, consider a firmware specification which does not completely list all the desirable capabilities of the computer instruction set. This lack of information may result in a hardware function design that is insufficient to support the desired instruction characteristics. But, it is also likely that the hardware function will be overspecified so that the desired instruction characteristics can be accommodated anyway.

The influence of hardware function design on hardware design is exceptional among the horizontal influences in that it tends to be stronger than the others. For digital functions the interconnections of digital logic must reflect the proper boolean functions. This implies restrictions on wire length, routing of interconnections, stray capacitance, etc. For requirements such as operator accessibility, mechanical interfaces with other systems, cooling, etc., the form of the hardware is strongly determined by the function that it must perform.

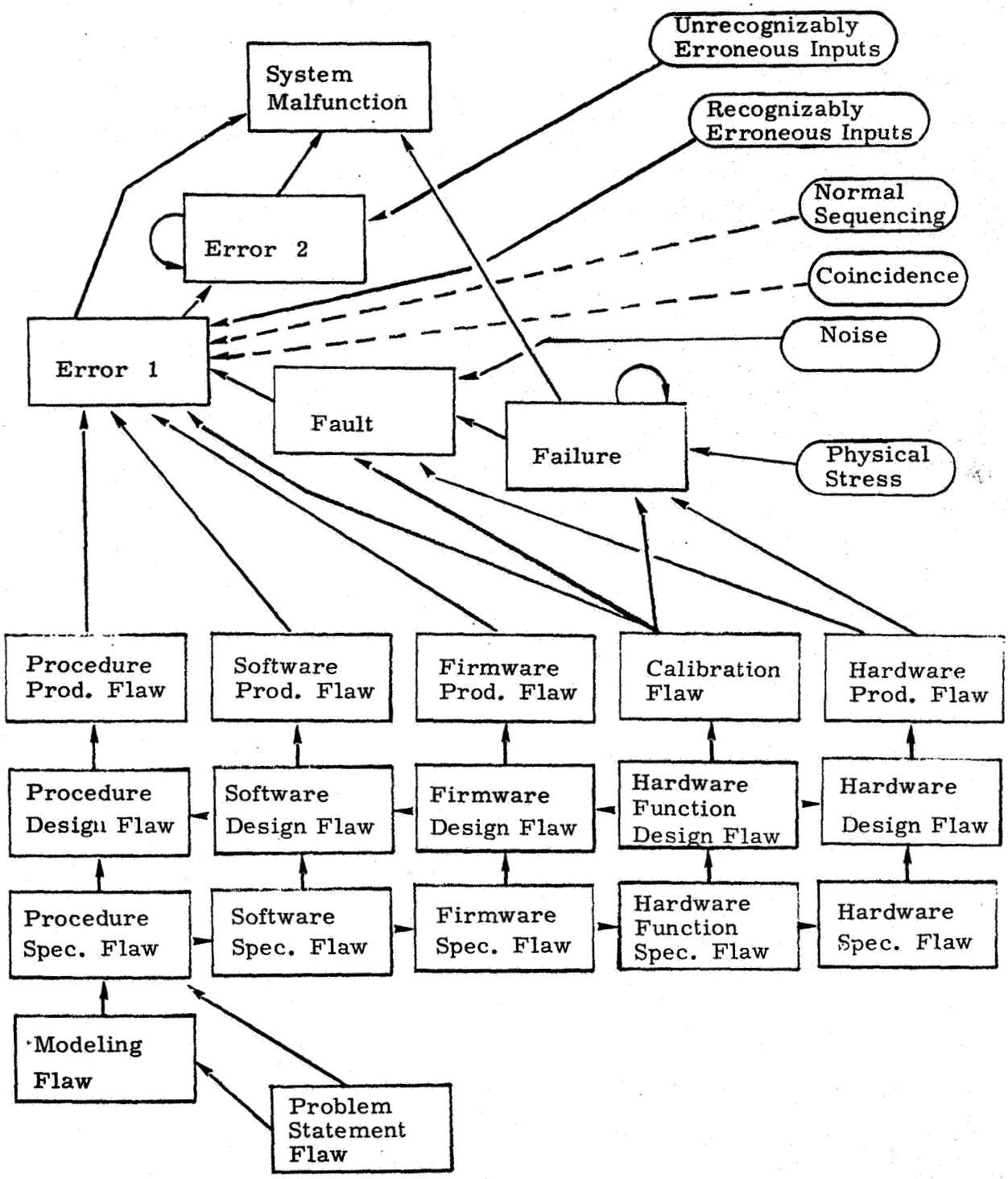
In order for a flaw to result in erroneous system behavior it must propagate to some production activity and thus be reflected in that activity. Fig. 3.5 shows the relationship between the development structure and the modes of erroneous behavior in the completed system. Notice that flaws reflected in procedure, software, and firmware production act to produce errors. Flaws reflected in hardware production act to produce faults and failures. Calibration implies an adjustment, or 'fine-tuning' of hardware. A flaw reflected in calibration can have many effects depending on the particular situation where it occurs. A mis-calibrated sensor will cause data to be in error; mis-calibration of the threshold of a noise filter will be responsible for faults, while an improper adjustment in an over-voltage protection circuit will cause failure of hardware.

Even in the simplified model of Fig. 3.5, the number of paths which a flaw can follow to a production activity is large. From procedure specification, alone, there are 12 paths to production activities. Anything which simplifies the structure will be advantageous, so ultimately, the least influential of the influences (mostly horizontal) may have to be ignored.

3.5 The Z-Dimension

Redundancy is introduced into the system development model by expanding the structure of Fig. 3.5 in the Z-dimension, out of the plane of the paper. This is done by creating as many copies of the structure as there are simplex elements of the system, and then merging the activities that are common to the various simplex elements (noting that an activity is not necessarily common to all elements).

A flaw that occurs in an activity which is common to more than one simplex element may ultimately be reflected in the production of each of those elements. The claim is that the existence of a flaw in such a common activity implies a statistical correlation of the failure probabilities of the affected simplex elements. A correlation of this sort is de facto—it exists because of the malfunction model being employed. The model states that



System Malfunction Model

Fig. 3.5

a malfunction occurs because an existing flaw is externally activated. Correlation comes about because redundant systems generally operate in proximity to one another and often operate simultaneously, so they are very likely to encounter the same flaw activating conditions. Moving the systems apart (in both space and time) decreases the likelihood of them encountering the same flaw activating conditions (and therefore decreases the correlation), but it does not eliminate the possibility of both systems encountering these conditions.

There is another type of correlation that can exist between simplex elements. This is due to a causal interaction of simplex elements during system operation. For example, if a prime element updates a backup element, and a flaw in the update operation allows the passage of bad data, then a failure of the prime element may leave a less than perfect backup. This type of behavior is like that of a system malfunction of one system acting as an external influence to another system. When considering simplex elements of a redundant system, this effect can be modeled as Z-dimensional interactions of error, failure and fault mechanisms.

Both forms of system correlation can be studied within the framework of the system development and analysis structure which has been discussed. This will allow an a priori prediction of correlation factors for a redundant system which can then be used to modify the reliability equation for that system. However, before considering the mathematical aspects of correlation, it is useful to investigate the basic structures for system redundancy and their associated flaw propagation structures.

3.6 Basic Structures for Redundancy

There are three basic ways in which redundancy can be employed at a system level.

1. Parallel with active backup—where one system is designated prime and the rest are kept updated and ready to take over immediately should the prime fail.

2. Parallel with passive backup—where one system is designated prime and the rest are dormant. In the event that the prime system fails, one of the dormant backups must be activated and initialized ("brought up to speed") before it can take over.

3. N-Modular Redundancy—where a majority vote of an odd number of active systems determines the overall system action. Failures in up to (N-1)/2 systems are masked.

In addition, these techniques can be used in combination, such as a two-out-of-three vote with spares to replace failed elements in the voting circuitry or 'hard core'.

Parallel structures do not require that the simplex elements be similar. The backup system(s) may be entirely different from the prime. It could be that the prime and backup systems have no more in common than minimal procedural specification and design. This much is necessary in order to activate the backup system should the prime fail. On the other hand, software and procedural specification and design activities could be common and still result in two rather different systems. It should be clear that for an activity to be considered "common" to two systems, it is not just a question of the output of the activity, but also "who" is doing it, and "where and how" it is being done. The same design group doing software for prime and backup systems implies the software design for the two systems will have something in common.

For completely identical systems configured in parallel, the entire development structure is the same. There is effectively only one copy of the algorithms for such a system, even though there is duplication of the physical hardware. Some of the 'sameness' can be avoided by designing

several programs to do a particular applications task and by designing workaround procedures in case a primary procedure should fail. However, there are some system aspects, hardware function, firmware and operating system software, which cannot easily be duplicated in this manner. The simplex elements of the system correlate highly at these points of common algorithm and the only defense against a malfunction, given this choice of system configuration, is to minimize the flaw densities of the associated activities.

N-Modular Redundant (NMR) systems fall prey to de facto correlations in the same way that identical parallel systems do, but the situation can be considerably worse. Whereas a parallel organization has some kind of algorithmically implemented self-error-detection capability, an NMR system usually does not. It relies on the voting scheme to detect and mask errors. An NMR system that is synchronized to the logic level offers almost no protection against malfunctions due to algorithmic flaws. All copies of the system are affected simultaneously and the voter does not even detect a disagreement. If the systems are loosely synchronized and the voting is done by software, it is possible, due to timing phenomena (coincidental circumstances), for an algorithmic flaw to be activated in some of the systems. Now the voter can do its work. Within certain limits it can be said that a decrease in synchronization implies a decrease in correlation.

Parallel system elements in which information is transferred between copies are also susceptible to causal correlations. This occurs in a configuration where the prime element is called upon to update the backup elements. A flaw in the prime element could result in bad data being passed to the backup. If data in the prime element is used to bring the backup up to speed after the prime has failed, then the correlation between the elements is even greater than before.

3.7 Systematic Failures

'Systematic Failures' has become a common term which is applied rather loosely to describe the kind of correlated malfunction modes that have been discussed. The term describes a situation in which a redundant system fails because a critical number of the simplex elements are afflicted with the same malfunction, or a situation in which the malfunction of one element causes the malfunction of the others. This is consistent with the notion of correlations being responsible for this type of behavior. De facto correlations increase the likelihood that simplex elements will be afflicted in the same way, and causal correlations make it possible for one element to bring down another element.

It is important to remember that correlation does not cause systematic failures. It is merely a way of describing the statistical behavior of the system. A positive correlation in the failure probability of two system elements implies that when one has failed, the probability of the second failing is greater than it would have been had the first not failed. Therefore, the second element is less effective in this case than in the case where the elements are independent and failure of the first implies nothing about the second.

Avionics designers tend to associate systematic failures primarily with electronic systems, because of the complicated electrical interconnections and high level algorithms. It should be pointed out that other types of systems are also very susceptible to this problem. Consider the case of the F-14 hydraulic systems which failed during flight test.[1] Titanium hydraulic lines burst almost simultaneously in identical prime and backup systems due to bending stresses. The second backup, which was different from the other two systems was incapable of controlling the plane under the circumstances and a crash ensued. The prime and first backup were almost completely correlated for this flaw. In effect, the major source of system redundancy was non-existent. The secondary redundancy was unable

to cope with the problem because of some other flaw in its development. This incident demonstrates not only that systems other than electronic are susceptible to systematic failures, but also just how insidious such failure modes are.

Conventional calculations of redundancy provide a reliability figure based on configurational analysis, assuming statistical independence of the simplex elements. Systematic failure modes make this assumption invalid. In order to avoid false confidence in a redundant system it is necessary to determine a measure of the effects of correlations on the reliability of the overall system. Some approaches to measuring these effects are discussed in Chapter 4.

CHAPTER 4

MATHEMATICS OF CORRELATION

4.1 Motivation

In order to quantitively assess the impact of systematic failure modes on a redundant system, it is necessary to include in the reliability equation the effects of correlations among simplex elements of the system. The way that this should be done depends upon the depth of information available, i.e., how well the correlations are known. The benefits of a complex formulation that accurately reflects all the n-wise correlations among system elements cannot be exploited fully if the correlations are only roughly known; a simpler approximation will give an answer that is just as good. On the other hand, if a more accurate knowledge of the correlations is available it should be employed to the maximum extent.

This chapter studies three approaches to including correlations in the reliability equation for a redundant system. The presentation of these approaches employs some common preliminary material which is given below.

4.1.1 Reliability

Reliability, denoted as R(t), is the probability that a system will not fail at or before a given time, t. The complement of the reliability function is called the failure distribution function, F(t).

$$R(t) = 1 - F(t) \tag{4.1}$$

F(t) is the probability that a system will fail at or before a given time, t.

The time of system failure, t, is a random variable that assumes each of its possible values with probability f(t), the failure probability density function. By the rules of probability $\int_{-\infty}^{\infty} f(t)dt = 1$. This merely states that t must assume one of its possible values. F(t) is defined in terms of f(t) as follows,

$$F(t) = \int_{-\infty}^{t} f(t)dt \tag{4.2}$$

Hence F(t) is the probability that the random variable t assumes a value less than or equal to t. Or, as stated above, F(t) is the probability that the system will fail at or before t.

For a real system that is working at t=0, the integral in (4.2) can be taken from zero to t, since the probability of failure before t=0 is clearly zero. F(t) increases monotonically from zero at t=0 to one at some t>0. Usually F(t) approaches one asymptotically as t approaches infinity. R(t), therefore, starts out equal to one at t=0 and eventually falls to zero.

Assume a simplex system element to have a reliability $R_u(t)$, and assume that all such elements are statistically independent. The reliability of a redundant configuration of these elements can be expressed as $g(R_u(t),t)$. The form of g is determined by the configuration of the elements.[2,7] For simplicity the t dependence of R_u will be dropped, as it is implicit and can be reinserted any time.

The explicit dependence of g on time appears only in certain standby redundancy configurations. Most of the analysis in the following sections employs active parallel and NMR configurations to simplify the derivations. However, the methods developed will be valid even if g is an explicit function of time.

Removing the time dependence from the equations (or fixing the value of time—depending on your point of view) reduces the problem from the analysis of continuous random variables to the analysis of discrete random

variables. This allows a conceptual development of the problem without a great deal of complicated mathematics.

4.1.2 Correlation

Mathematical correlation is a way of expressing a statistical dependence among a group of random variables. When two random variables exhibit a dependence, then knowledge of the value of one increases knowledge of the value of the other. If the random variables are only slightly dependent, then the ability to predict the value of the second given the value of the first is not much greater than the ability to predict the value of the second without knowledge of the value of the first. If the random variables are completely dependent, then the value of the second can be determined exactly given the value of the first.

In the literature of statistics the word correlation is precisely defined to mean the joint expectation of a set of random variables. The most common measure of correlation is called the coefficient of linear correlation, ρ. It is defined as the normalized covariance of a set of random variables.

$$\rho = \frac{\text{cov}(S_1, S_2, \ldots, S_n)}{\sigma(S_1)\sigma(S_2)\ldots\sigma(S_n)} \tag{4.3}$$

Where $\text{cov}(S_1, S_2, \ldots, S_n)$ is the joint expectation of $S_1, S_2, \ldots, S_n$ taken about their respective means, and $\sigma(S_1), \sigma(S_2), \ldots, \sigma(S_n)$ are the standard deviations of $S_1, S_2, \ldots, S_n$ respectively.

No attempt will be made here to redevelop the theories of random variables or linear prediction. Reference 13 contains excellent discussions of all the concepts that are needed to follow the ensuing arguments. Suffice it to say that the notion of correlation developed intuitively in Chapters 2, 3 and 4 is not inconsistent with the accepted definitions. The word "correlation" will still be used to imply both the phenomenon and the

measure of that phenomenon. The measure will be seen to be the same, under certain conditions, as the coefficient of linear correlation, ρ.

The scope of a correlation is defined as the number of elements that are jointly correlated. A correlation of four elements is of greater scope than a correlation of only three of those elements. Correlation does not have transitive properties. A correlates with B, and B correlates with C, does not imply A correlates with C. In fact, even if A does correlate with C, the three pairwise correlations do not imply that A, B and C are jointly correlated. There are certain relationships which can be developed between correlations of different scope, where the elements of one correlation are a subset of the elements of the other correlation. These are discussed in Section 4.3.6.

4.1.3 Approaches to Including Correlation in the Reliability Equation

The approaches to including correlations in the reliability equation are presented in the order in which they were developed. This allows the timely presentation of some of the problems that were encountered.

The first approach (Sec. 4.2) considers how the reliability improvement gained by the use of redundancy is affected by correlations among the simplex system elements. The overall system reliability is expressed in terms of a reliability improvement measure, the reliability of the simplex elements, and the correlation. This is done for two different measures of reliability improvement, and the relationship between the two formulations is investigated. The equations are straightforward, but they are not exact unless all of the correlations among the the simplex elements can be expressed as a single parameter.

The second approach (Sec. 4.3) returns to the basics of joint probability distributions to derive a reliability equation that is exact. The correlations among simplex system elements are considered without regard to the flaws which cause them. The problem with this approach is that the

algebra becomes very complex for situations other than those in which all simplex elements are jointly correlated, i.e., situations in which the correlations can be expressed as a single parameter.

The third approach (Sec. 4.4) approach considers correlations in the light of the flaws that cause them. By properly decomposing the reliability expression for each simplex element, the overall system reliability can be written in such a way that the simple expressions for joint correlation derived in Sec. 4.3 can be successfully applied.

4.2 Reliability Improvement Approach

Reliability improvement measures are used to assess the benefits obtained by the application of redundancy to a simplex system. Cost-performance tradeoff analyses use such measures to determine if the added expense of redundancy provides a high enough return in reliability. Reliability improvement measures can also be used to compare the effectiveness of various redundancy schemes.

The role of reliability improvement measures in a discussion of systematic failures is derived from the following assumption. If the simplex elements of a redundant system are correlated, then a reliability improvement measure ought to reflect less improvement than it would if the simplex elements were not correlated. When the elements of a system are completely correlated, the system reliability should be the same as the reliability of one of the simplex elements. When the elements are completely independent, the reliability of the system should be the maximum afforded by the particular redundancy configuration.

4.2.1 Additive Measure

The maximum system reliability for a given configuration can be expressed as a function of the reliability of a simplex element and the

reliability improvement measure. The effects of correlations can be included in the reliability expression by defining a function which can be used to scale the reliability improvement. Such a function of correlation will be called an effectiveness factor, for it reflects how effective the redundancy actually is in producing an improvement in reliability.

If the reliability of an unredundant system element is equal to R_u, and the reliability of the redundant system can be expressed as $R_r = g(R_u)$, then the additive reliability improvement is defined as

$$I(R_u) = g(R_u) - R_u \tag{4.4}$$

The redundant system reliability, therefore, is

$$R_r = I(R_u) + R_u \tag{4.5}$$

The effectiveness factor, E(c), scales $I(R_u)$ so that $E(c)I(R_u) = 0$ when the simplex elements are completely correlated, and $E(c)I(R_u) = I(R_u)$ when the simplex elements are independent. The equation for the actual system reliability, R, is

$$R = E(c)I(R_u) + R_u \tag{4.6}$$

where $0 \leqslant E(c) \leqslant 1$. If c, the measure of correlation, varies from 0, for complete independence, to 1, for complete correlation, and if E(c) is assumed to be linear in c, then

$$E(c) = 1-c \tag{4.7}$$

The additive reliability improvement measure, $I(R_u)$, is not particularly useful for evaluating the quality of a redundant system configuration. I=0.2 could represent a much smaller improvement than I=0.009, depending upon the value of R_u. However, the additive improvement measure does

allow the effectiveness factor, E(c), to be included in the reliability equation in a straightforward manner. I.e., the "real" reliability improvement, $I'(R_u)$, is expressible as a function of correlation, E(c), times the "theoretical" reliability improvement, $I(R_u)$.

4.2.2 Ratio Measure

Another type of reliability improvement measure can be obtained by taking the ratio of the redundant system reliability to the simplex system reliability. A measure of this type has been proposed by T. Klaschka.[6] Her system organization assumes that

1. The unredundant system is composed of unredundant elementary segments taken in series, so the unredundant system reliability is the product of the reliabilities of each unredundant elementary segment.

2. The redundant system is composed of redundant elementary segments taken in series, so the redundant system reliability is the product of the reliabilities of each redundant elementary segment.

If a simple ratio were taken, the improvement index would be dependent on system size.

$$y = R_r^S / R_u^S$$

To eliminate this dependence, Klaschka takes the logarithms of both the redundant and the unredundant system reliabilities. Since $0 \leqslant R_u < R_r \leqslant 1$ implies $\log R_u > \log R_r$, the ratio is inverted so that y will remain greater than one. Hence,

$$y = \log(R_u^S)/\log(R_r^S) = s\log R_u / s\log R_r = \log R_u / \log R_r$$

so the reliability improvement measure is independent of system size.

Klaschka's measure has a drawback, with respect to the application under consideration, in that the logarithms will complicate the algebra. In addition, in this application, the reliability improvement index will not be used to evaluate the quality of a redundancy configuration, per se, so that the absolute magnitude of y is not of paramount importance. Therefore, a more convenient measure of this type can be obtained by taking the ratio of the failure probability of a simplex element, F_u, to the failure probability of the redundant system, F_r. If F_r can be expressed as $h(F_u)$, then y, the reliability improvement, is expressible as

$$y = F_u/F_r = F_u/h(F_u) = (1-R_u)/(1-g(R_u)) \tag{4.8}$$

The expression for y is the "theoretical" reliability improvement given no correlation among simplex system elements. If F(=1-R) is the "real" system failure probability, including correlation, then y', the actual reliability improvement factor, is given by

$$y' = F_u/F = (1-R_u)/(1-R) \tag{4.9}$$

So y' can be related to y by defining an appropriate effectiveness factor X(c). When there is no correlation (X(c)=0) then y'=y, the reliability improvement is the maximum theoretically possible; when there is total correlation (X(c)=1) then y'=1, and there is no reliability improvement. This function is expressed as

$$y' = y/[(y-1)X(c) + 1] \tag{4.10}$$

where

$$X(c) = c \tag{4.11}$$

In the additive reliability improvement case, the real improvement is expressible as the product of the theoretical improvement and the effectiveness factor, $I'(R_u) = E(c)I(R_u)$. From (4.10) it can be seen that this simple relationship does not exist for the ratio case.

From (4.8), (4.9), and (4.10) the probability of system failure is

$$F = [(y-1)X(c)+ 1]h(F_u) \tag{4.12}$$

An expression for reliability is obtained by replacing F with 1-R and replacing $h(F_u)$ with $1-g(R_u)$

$$R = [g(R_u)(y+1)-y]X(c) + y + g(R_u) \tag{4.13}$$

4.2.3 Evaluation

Equations (4.6) and (4.12) express redundant system reliability as a function of simplex element reliability and the correlation among the simplex elements. Equation (4.6) is algebraically simpler when discussing reliability (probability of system success), (4.12) is simpler when discussing probability of failure.

The two equations were derived using very different reliability improvement measures, but the equations are very similar from an applications viewpoint. Each relies on a single correlation coefficient, expressed as the effectiveness factors E(c) and X(c). E(c) and X(c) are closely related, E(c) = 1-X(c).

All of the different possible correlation relationships among the simplex system elements must be reduced to a single correlation figure. If correlations other than the joint correlation of all simplex elements exist, then they cannot be accurately reflected by (4.6) or (4.12). This approach to expressing correlation, however, is fairly simple and does provide a more accurate picture of the system reliability than if no correlation factors are included at all.

4.3 Joint Correlation Approach

In order to obtain an exact picture of the effect of correlations on reliability, it is necessary to develop equations that include all correlation factors. The object is to find a formulation which, though exact, is easy to apply in practice. The initial approach is to investigate the characteristics of some simple configurations and to see how these relate to the results of the previous section.

The initial development is restricted to active parallel redundancy schemes because they are the least complicated to analyze. An n-element parallel system where the elements are all statistically independent fails only if all of its n elements fail, so

$$F_{sys} = \prod_{i=1}^{n} F_i \tag{4.14}$$

For all other 2^n-1 combinations of element failure and success, the system succeeds.

Consider a system composed of two identical elements in parallel. If they are uncorrelated, then by (4.14), $F_{sys} = F^2$. If they are completely correlated then they always behave in the same manner, and it is as though there is only one copy of the element, so $F_{sys} = F$. For intermediate values of correlation, say 1/2, one might intuitively take a linear combination of the extremes of complete independence and complete correlation. This gives $F_{sys} = F/2 + F^2/2$, which generalizes to $F_{sys} = cF + (1-c)F^2$. In fact, this expression turns out to be correct, as the following development will show.

4.3.1 Joint Probability

The overall behavior of a set of system elements is described by the joint probability distribution for those elements. The joint distribution assigns a probability of occurrence to each possible state of the total system.

For simplicity consider a two-element system. If each element has two states (working or failed) then the system has four states. Each of these four states has a probability of occurrence, and since one of the four states must occur, the four probabilities must sum to one. Joint distributions are mathematically the same as regular distributions. They are singled out for special attention because they can be expressed in a form that makes it easy to see the statistical relationships among the system elements.

A system element will be characterized by a random variable S, the probability of success. The element fails (denoted by S^F or S=0) with probability F, and succeeds (denoted by S^S or S=1) with probability $\overline{F}$ (=1-F). This describes a probability distribution for the system element. In general, the joint probability distribution for a set of system elements cannot be obtained from the individual distributions. However in certain cases such a determination can be made. These cases of interest are when the elements are statistically independent and when they are completely correlated.

If S_1 and S_2 are independent then the probability of S_1 and S_2 both occurring is equal to the probability of S_1 occuring times the probability of S_2 occurring.

$$P(S_1 \cap S_2) = P(S_1)P(S_2) \tag{4.15}$$

The joint distribution for S_1 and S_2 independent is shown in Fig. 4.1.

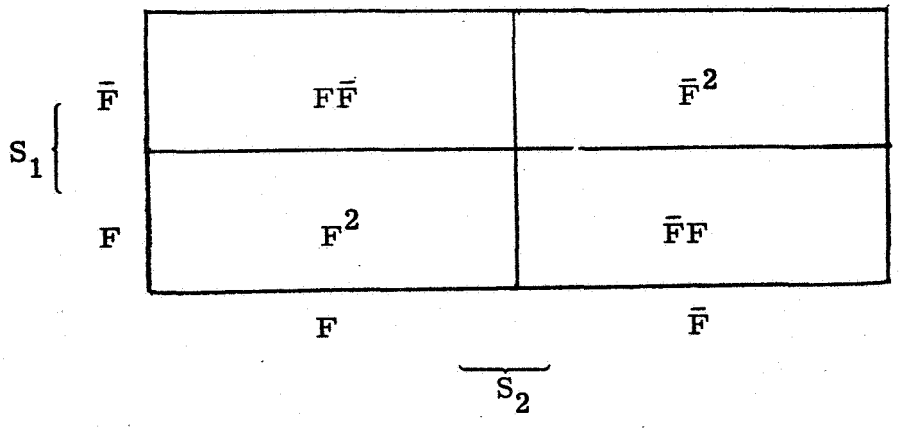


Fig. 4.1

Although the joint distribution cannot, in general, be found from the individual distributions, the individual distributions can always be extracted from the joint distribution by summing the appropriate rows or columns. Because the individual distributions appear in the margins of the joint distribution they are often referred to as marginal distributions.

If S_1 and S_2 are completely correlated, then they always behave in the same way. Those states where S_1 and S_2 are not the same have a zero probability of occurrence. In order to be able to recover the appropriate marginal distributions from the joint distribution, those states where $S_1 = S_2$ assume probabilities as in Fig. 4.2.

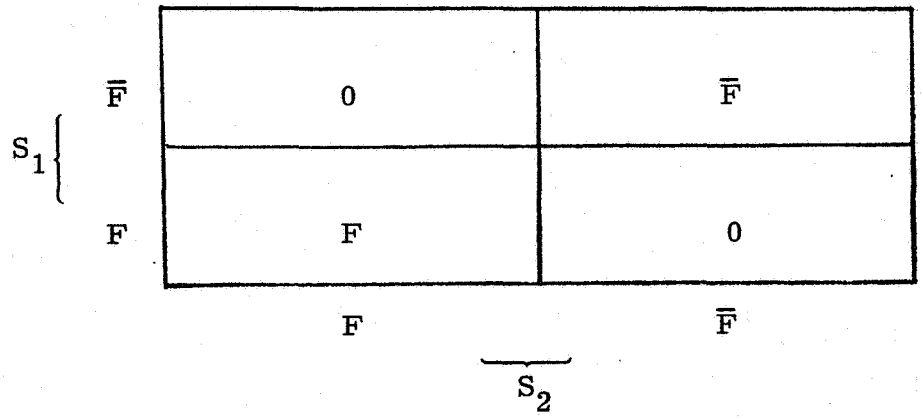


Fig. 4.2

It is desirable to develop a single general expression for the joint probability of S_1 and S_2, in which the correlation, c, represents the degree of dependence between S_1 and S_2. Such an expression can be developed using the concept of conditional probability. $P(S_1|S_2)$ denotes the conditional probability of S_1 given S_2. It represents the probability that S_1 will occur given that S_2 is known to have already occurred. $P(S_1|S_2)$ is defined as $P(S_1 \cap S_2)/P(S_2)$, so

$$P(S_1 \cap S_2) = P(S_1|S_2)P(S_2) \tag{4.16}$$

and by symmetry

$$P(S_1 \cap S_2) = P(S_2|S_1)P(S_1) \tag{4.17}$$

Each of the four terms of the joint probability distribution can be represented by an equation similar to (4.16).

Consider the term for S_1 and S_2 both failing, $P(S_1^F \cap S_2^F) = P(S_1^F|S_2^F)P(S_2^F)$. (This happens to be the same as (4.16).) If the systems are independent and if S_1 fails with probability F, then $P(S_1^F|S_2^F) = F$. If the systems are completely correlated then $P(S_1^F|S_2^F) = 1$. Fig. 4.3 is a plot of $P(S_1^F|S_2^F)$ vs. c.

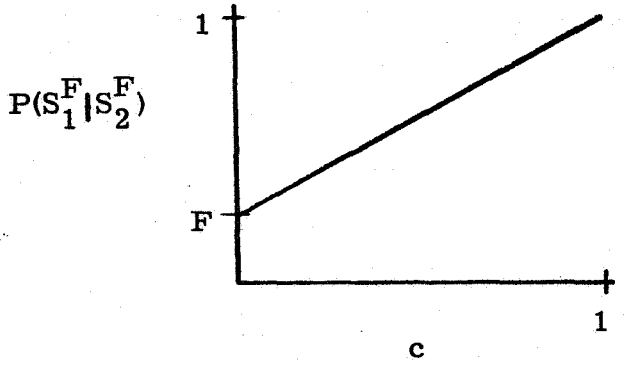


Fig. 4.3

The probability of S_1 and S_2 both failing, stated as a function of c, is

$$P(S_1^F \cap S_2^F) = (c(1-F) + F)F = cF + (1-c)F^2 \tag{4.18}$$

The other terms of the distribution can be found in the same manner. For example, $P(S_1^F \cap S_2^S) = P(S_2^S|S_1^F)P(S_1^F)$. S_1 and S_2 independent implies that $P(S_2^S|S_1^F) = \overline{F}$. S_1 and S_2 correlated implies that $P(S_2^S|S_1^F) = 0$. The line generated by these two point is given in Fig. 4.4

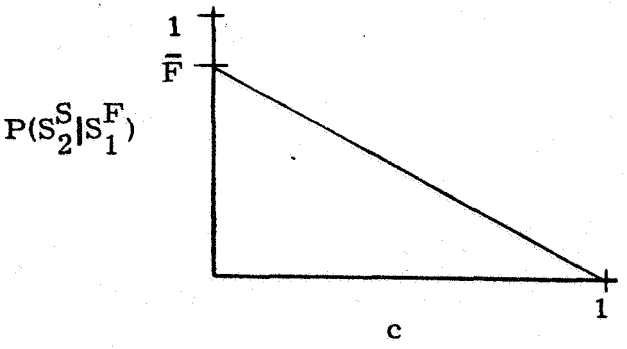


Fig. 4.4

So the probability for S_1 failing and S_2 succeeding is given by

$$P(S_1^F \cap S_2^S) = (1-c)\bar{F}F \tag{4.19}$$

By symmetry with (4.18) and (4.19)

$$P(S_1^S \cap S_2^F) = (1-c)\bar{F}F \tag{4.20}$$

$$P(S_1^S \cap S_2^S) = c\bar{F} + (1-c)\bar{F}^2 \tag{4.21}$$

The entire joint distribution expressed as a function of correlation is shown in Fig. 4.5.

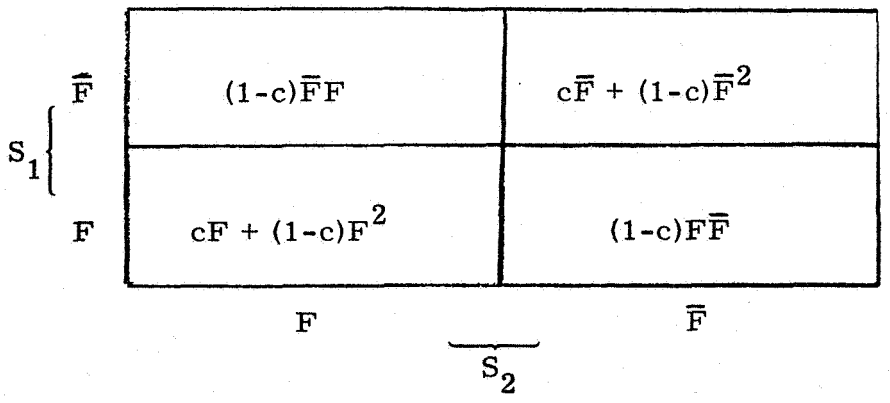


Fig. 4.5

This is a valid distribution since the rows and columns sum to the appropriate marginal probabilities.

The joint distribution can be computed without requiring the factorization of each term as in (4.16). Plotting the extremes of the joint distributions (independence and complete correlation) vs. c will produce the above results quite easily. This is useful in the following situation where S_1 and S_2 do not have the same marginal distributions.

Consider two systems S_1 and S_2 where the probability of failure of S_1 is F_1, the probability of failure of S_2 is F_2, and $F_2 < F_1$. If the systems are independent then the joint distribution is given by Fig. 4.6.

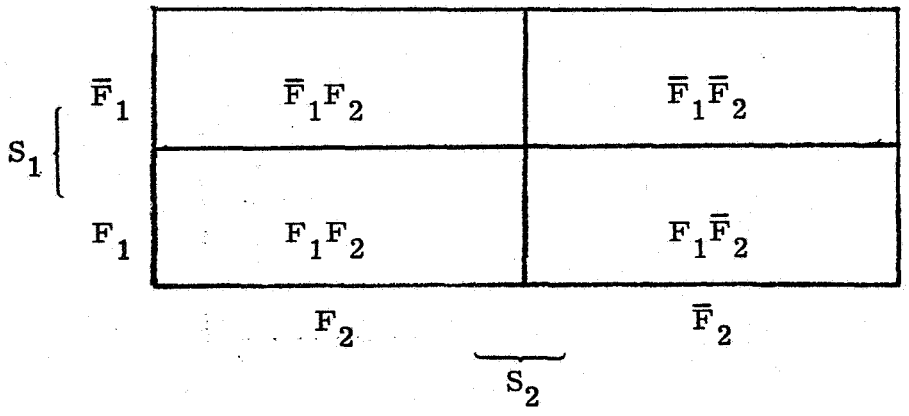


Fig. 4.6

If the systems are completely correlated, then S_2^F implies S_1^F because $F_2 < F_1$; therefore $P(S_2^F \cap S_1^S) = 0$. The rest of the joint distribution can be found by satisfying the requirement that the appropriate rows and columns sum to the marginal distributions. Thus the joint distribution for S_1 and S_2 dependent is shown in fig. 4.7.

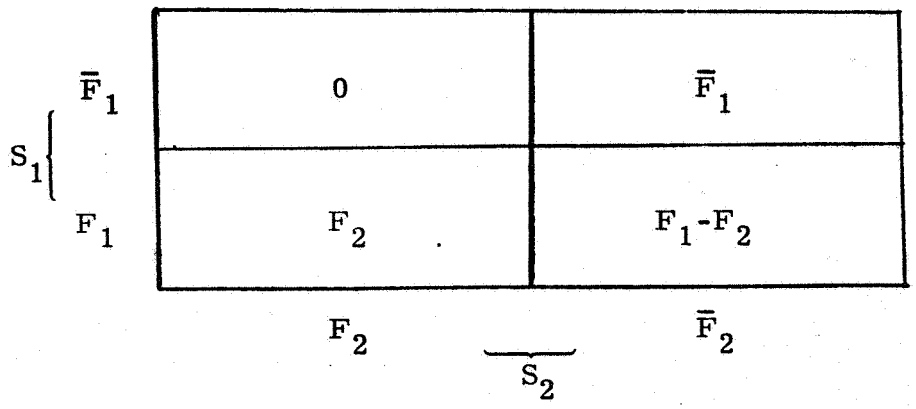


Fig. 4.7

The general form of the joint distribution as a function of c is shown in Fig. 4.8.

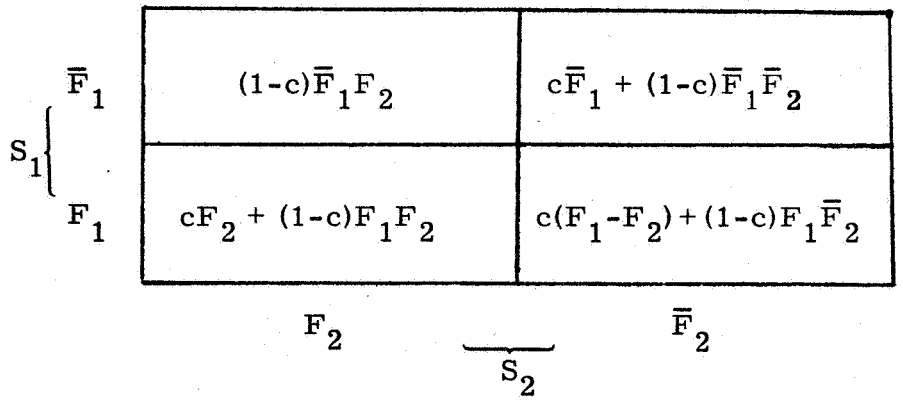


Fig. 4.8

The four terms add to 1, as they should.

4.3.2 Coefficient of Statistical Correlation

This section will relate c to the coefficient of linear correlation, ρ, which was discussed earlier. A system can be described by a random

variable S_i which has the following characteristics ($\overline{F}_i$ in this case indicates the complement of F_i):

1) System success, $S_i=1$, occurs with probability $\overline{F}_i$.

2) System failure, $S_i=0$, occurs with probability F_i.

3) The expected (mean) value of S_i is equal to $\overline{F}_i$.

For two systems described by random variables S_1 and S_2, is defined as

$$\rho = \text{cov}(S_1,S_2)/\sigma(S_1)\sigma(S_2) \tag{4.22}$$

The covariance is the joint expectation of S_1 and S_2 taken about their respective means, M_1 and M_2.

$$\text{cov}(S_1,S_2) = \text{Expectation}[(S_1-M_1)(S_2-M_2)] \tag{4.23}$$

If $F_1 = F_2 = F$, then

$$\begin{aligned}\text{cov}(S_1,S_2) &= \sum_{S_1}\sum_{S_2} P(S_1 \cap S_2)(S_1-\overline{F})(S_2-\overline{F}) \\ &= (cF + (1-c)F^2)(-\overline{F})(-\overline{F}) \\ &\quad + 2(1-c)F\overline{F}(-\overline{F})(1-\overline{F}) \\ &\quad + (c\overline{F} + (1-c)\overline{F}^2)(1-\overline{F})(1-\overline{F}) \\ &= cF\overline{F}\end{aligned} \tag{4.24}$$

Since $\sigma(S_i)$, the standard deviation of S_i, equals $\sqrt{F\overline{F}}$ for both S_1 and S_2,

$$\sigma(S_1)\sigma(S_2) = F\overline{F}$$

Therefore $\rho = c$. This is an interesting result for it says that ρ is independent

of the failure probabilities of the two systems. However, c corresponds to ρ only for the case where $F_1 = F_2$.

For the case where F_1 and F_2 are not equal—specifically where $F_1 = KF_2$, $K>1$

$$\rho = c\sqrt{(1-F_1)/(K-F_1)} \tag{4.25}$$

This more general result shows that the coefficient of linear correlation, ρ, is never more than c, the correlation factor which appears in the joint distributions. It can be considerably less, however, depending on the values of F_1 and F_2.

The fact that c does not always correspond to ρ suggests that it measures something different. In fact, c is a measure of the mechanisms that are responsible for correlated malfunctions rather than a measure of the statistical correlation itself. This follows if one considers the way in which c was defined for the joint distribution where $F_1 \neq F_2$. Recall that with c=1, $S_2^F \Rightarrow S_1^F$, (but the converse is not always true). What this says is that all of the failure mechanisms of S_2 exist in S_1 and that these particular mechanisms are completely correlated between S_2 and S_1. There are, however, failure mechanisms in S_1 which do not exist in S_2, so S_1 can fail without S_2 failing. The ability to attribute correlation to particular failure mechanisms is more fully exploited in Sec. 4.4.

4.3.3 Relation to Reliability Improvement

This section will discuss the relation between the reliability improvement approach to expressing system reliability and the joint probability approach developed in Sec. 4.3.1. The comparison is done for two system elements in parallel. This sets the stage for the problems encountered in expanding to more than two elements.

The joint probability expression for system failure, $F = cF_u + (1-c)F_u^2$, can be put in the additive reliability improvement form, $R = E(c)I(R_u) + R_u$ by replacing F with 1-R and F_u with $1-R_u$.

$$1-R = c(1-R_u) + (1-c)(1-R_u)^2$$
$$R = (1-c)[R_u - R_u^2] + R_u$$
$$R = (1-c)[(1-(1-R_u)^2) - R_u] + R_u$$

Since this is a two-element parallel system, $(1-(1-R_u)^2)$ is equal to R_r, the "theoretical" redundant system reliability. Therefore,

$$R = (1-c)I(R_u) + R_u \tag{4.26}$$

and $E(c) = 1-c$ as it did in (4.7).

A similar manipulation can be done for a system in which the reliabilities of the two elements are unequal. Consider two elements with reliabilities R_1 and R_2, where $R_2 > R_1$. Using the relationships

$$R = 1-F$$
$$R_1 = 1-F_1$$
$$R_2 = 1-F_2$$
$$R_r = g(R_1, R_2) = 1-(1-R_1)(1-R_2)$$

the equation for system failure, $F = cF_2 + (1-c)F_1F_2$, can be put into the reliability improvement form by substitution.

$$1-R = c(1-R_2) + (1-c)(1-R_1)(1-R_2)$$
$$R = (1-c)[R_1 - R_1R_2] + R_2$$
$$R = (1-c)[1-(1-R_1)(1-R_2)-R_2] + R_2 \tag{4.27}$$

Again it is seen that $E(c) = 1-c$.

Relationships between the joint probability formulation and the ratio reliability improvement formulation can be shown in a manner similar to the above.

It is not surprising that the above relationships exist since both approaches use some common assumptions in their development. First, the correlations are presupposed to be independent of the reliabilities of the individual system elements; and second, the correlations are assumed to be joint over all of the redundant system elements. The first assumption, that of independence, is a matter of definition. Although may not be independent of the system element reliabilities, c is defined to be so. The second situation, that only a joint correlation of all the system elements is present, occurs de facto—only two-element systems have been discussed. Complexities arise when systems with more than two elements are considered.

4.3.4 Systems with More than Two Elements

Two-element systems are not very complex. The only configurations that make any sense have the two elements in parallel, where the backup unit may be either active or passive. Voting schemes must obviously be ruled out because they require at least three elements. Given F_1, F_2 and c, it is possible to analyze the permissible two-element configurations quite thoroughly.

Parallel systems with more than two elements are straightforward if the only correlation present is the joint correlation of all system elements. For example, in a three-element parallel configuration the probability of system failure is given by the term where all three elements fail. If the elements are independent then $F = F_u^3$. If the elements exhibit total joint correlation then $F = F_u$. The same form holds as in the two-element case, so $F = cF_u + (1-c)F_u^3$. This generalizes to N elements as

$$F = cF_u + (1-c)F_u^N \tag{4.28}$$

The generalization is valid as long as c is joint across all the system elements, even if they do not all have the same failure probability. For the case of unequal failure probabilities in the two elements the equation becomes

$$F = cF_i^* + (1-c)\prod_{i=1}^{N} F_i \tag{4.29}$$

where F_i^* is the smallest of the F_i's. This follows from the discussion in Sec. 4.3.1.

In a system of three or more elements it is possible to have correlations among subsets of the elements. Introducing these subset correlations into the reliability equation by means of joint probability proves to be too complex an approach to obtain a general solution easily. The difficulties are circumvented by the techniques of Sec. 4.4. Some of the effects of subset correlations are shown in the following two sections.

4.3.5 Correlations among Subsets of Elements

Consider a three-element parallel system where S_1 and S_2 are correlated with each other but not with S_3. The failure probability of S_1 and S_2 is

$$F_{12} = c_{12}F + (1-c_{12})F^2 \tag{4.30}$$

Since S_3 is independent of the other two elements, it is included in the equation by simple multiplication.

$$F_{123} = (c_{12}F + (1-c_{12})F^2)F$$

$$F_{123} = c_{12}F^2 + (1-c_{12})F^3 \tag{4.31}$$

The general result can be obtained for an N-element parallel system where a <u>single</u> subset of m elements exhibits joint correlation, c_m.

$$F_{sys} = (c_m F + (1-c_m)F^m)F^{m-N} \qquad (4.32)$$

If more than one subset of elements exhibits correlation then the simplicity of (4.32) breaks down.

At this point it can be seen how correlations over proper subsets of elements affect the Reliability Improvement Approach of Sec. 4.2. Transforming (4.31) into the ratio reliability improvement form and solving for X(c) gives

$$cF^2 + (1-c)F^3 = ((1/F^2 - 1)X(c) + 1)F^3$$

$$X(c) = c(F/(1+F)) \qquad (4.33)$$

and since E(c) = 1-X(c),

$$E(c) = 1-c + c/(2-R) \qquad (4.34)$$

This serves as a counter-example to show that E(c) and X(c) cannot always be expressed as functions of c alone.

The basic inadequacy of the Reliability Improvement Approach is that it cannot, within its limited form, take into account correlations over proper subsets of system elements. This is not, however, sufficient reason to disregard the approach altogether. It is useful even though it has limited applicability.

4.3.6 Relations of Scope

Scope refers to the size of a subset of elements which exhibit a correlation. A subset of three elements has a greater scope than a subset of two elements.

Consider a three-element system with correlations c_{12}, c_{23}, c_{13} and c_{123}. One restriction on the pairwise correlations is that $c_{ij} \geq c_{jk} + c_{ki} - 1$. For example, if S_1 correlates .6 with S_2, and if S_2 correlates .6 with S_3, then S_1 must correlate at least .2 with S_3. A second relationship says that if an element correlates with subsets of different scope then the sum of such correlations must not exceed one, e.g., $c_{ij} + c_{123} \leq 1$.

The basis for these relationships comes from the fact that systems exhibit correlated failure behavior because of common failure mechanisms in the systems. The relationships are obtained by dividing each element of the system into failure mechanisms and assigning all possible correlations. This is depicted graphically in Fig. 4.9.

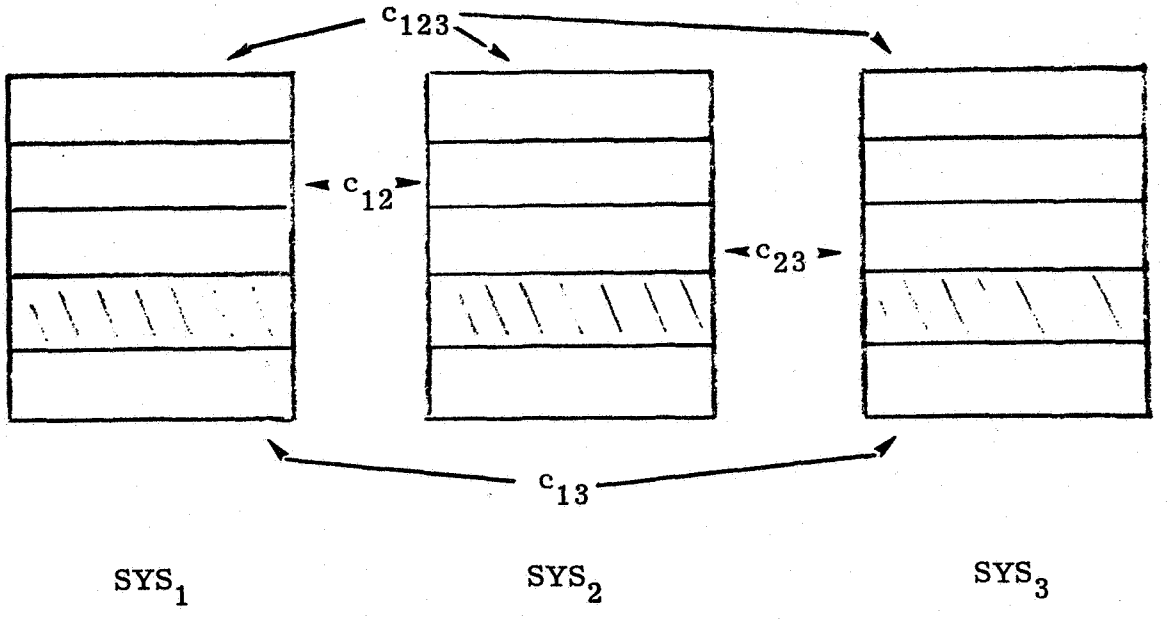


Fig. 4.9

Fig. 4.9 implies a decomposition of each element into parts, where the various modes of correlation relate particular mechanisms in the different systems. However, in the joint probability analysis, which views an element as a whole, the correlations are not assignable to mechanisms

in each element, they serve only to relate the statistical behavior of the elements without regard to the causes of that behavior. It is difficult, if not impossible, to derive the correlations by observation of the failure behavior of the elements. Without an analysis of the causes of failure, the effect of a correlation of a certain scope cannot be distinguished from the effect of several correlations of a lesser scope. That is, failures that look like they are related by c_{123} may bear the relation of c_{12} and c_{23}.

4.3.7 Evaluation

The Joint Probability Approach to including the effects of correlations in the reliability equation is practical only under certain circumstances. In general, the cases to which it can be easily applied are two-element systems, and systems of more than two elements in which there is a correlation over a single subset of the elements. These two cases, however, serve as important tools for implementing the failure mechanism decomposition described in Sec. 4.4.

4.4 Failure Mechanism Approach

This approach to including the effect of correlation in the reliability equation considers the mechanisms which cause failures. The failure probability of each element is decomposed into a product of the probabilities of occurrence of the failure mechanisms associated with that element. Knowledge of the system development structure can then be used to determine the correlations that exist among the failure mechanisms of the various elements. The reliability equations can be generated in a fairly simple and straightforward manner without worry about the reconciliation of correlation coefficients. As in the previous approach, the first step is to examine a system with two elements in parallel.

4.4.1 Relationship with Joint Probability Representation

Consider a system composed of two identical elements with failure probability F and which are correlated c. The failure probability, F, can

be expressed as two parts: one part which is related to correlated failures, F_B, and another part which is related to independent failures, F_A. Fig. 4.10 depicts graphically that system failure occurs if there is no path from X to Y.

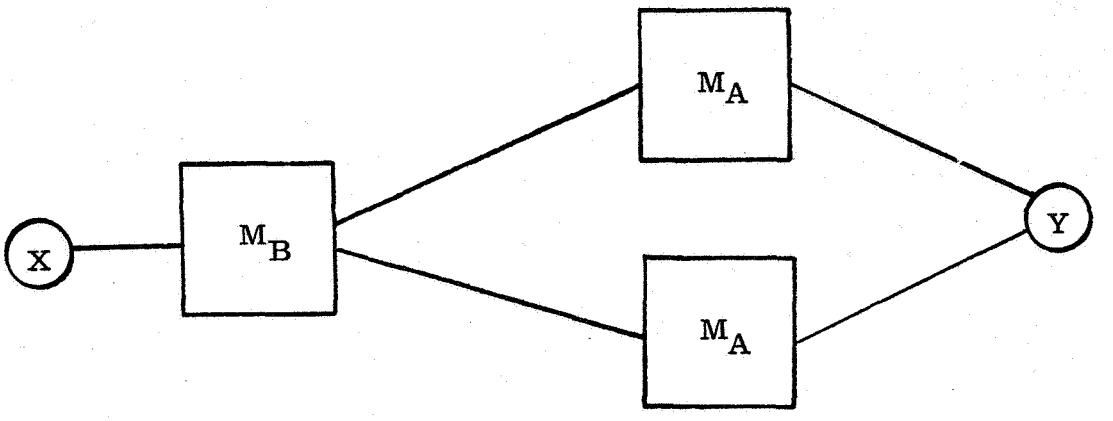


Fig. 4.10

If M_A and M_B are considered to be failure events with probabilities of occurrence F_A and F_B respectively, then the probability of no path from X to Y is just

$$F_{sys} = F_B + F_A^2(1-F_B) \tag{4.35}$$

F_B can be thought of as the probability of occurrence of those failure mechanisms which are completely correlated in the two elements, and F_A the probability of occurrence of those failure mechanisms which are completely independent in the two elements.

F_A and F_B can be found by examining the relationship between the joint probability form of F_{sys}, and the failure mechanism form of F_{sys}.

$$F_{sys} = cF + (1-c)F^2 = F_B + F_A^2(1-F_B) \tag{4.36}$$

This is an equation in four variables. To get F_A and F_B in terms of F and c, another equation is needed.

Consider the relationship of F_A, F_B, and F for a single element. They represent the probabilities of occurrence of events M_A, M_B and element fail, respectively. The element fails if either M_A or M_B occurs. Alternatively, the element does not fail if neither M_A nor M_B occurs. This is expressed as

$$1-F = (1-F_A)(1-F_B)$$

$$F = F_A + F_B - F_A F_B \tag{4.37}$$

Mathmatically this implies that each element can be decomposed into two modules which are in series. The modules represent groups of mutually exclusive failure mechanisms of an element. The A modules of each element are completely independent; the B modules are totally correlated. Using (4.37) to substitute for F_B in (4.36) gives

$$F^2(1-c) + F(c-1-F_A) + F_A = 0$$

which can be solved for F_A.

$$F_A = (1-c)F \tag{4.38}$$

Substituting into (4.37) gives

$$F_B = cF/(1-(1-c)F) \tag{4.39}$$

Equations (4.38) and (4.39) give the relationships for expressing a correlation between two system elements in terms of failure mechanisn modules, where the failure mechanisms which form one of the modules are independent and the failure mechanisms which form the other module are totally correlated.

4.4.2 Failure Mechanism Decomposition

The fact that a system element can be represented as a series of modules, each responsible for mutually exclusive failure mechanisms, allows the correlations to be put into the system reliability equation in a simple manner.

Consider the system of Sec. 4.4.1 which has two identical elements in parallel, except that here each element is composed of two modules in series. The failure probability of one module is F_A and that of the other is F_B. The system fails if there is no path from X to Y, Fig. 4.11.

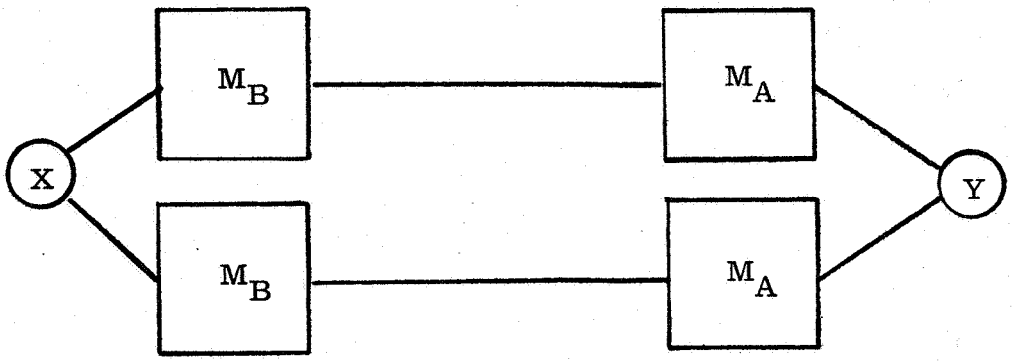


Fig. 4.11

From (4.37), the failure probability of each element is $F = F_A + F_B - F_A F_B$. Therefore the probability of system failure is

$$F_{sys} = F^2 = (F_A + F_B - F_A F_B)^2$$

$$F_{sys} = F_A^2 + F_B^2 + F_A^2 F_B^2 + 2F_A F_B - 2F_A^2 F_B - 2F_A F_B^2 \qquad (4.40)$$

Now consider correlation of the B modules. If they are independent, then $F_{sys} = F^2$. If they are completely correlated, then the probability that both fail is the same as the probability that one fails. Therefore, the value

of F_{sys} for complete correlation can be determined by replacing all F_B^2 terms in (4.40) with F_B. This gives

$$F_{sys} = F_B + F_A^2(1-F_B) \tag{4.41}$$

which is identical to the representation in (4.35).

Intermediate values of correlation in the B modules can be represented by replacing all F_B^2 terms in (4.40) with the expression for pairwise correlation of two elements, $cF_B + (1-c)F_B^2$. This gives

$$F_{sys} = F_A^2 + 2F_AF_B - 2F_A^2F_B + (cF_B + (1-c)F_B^2)(1 + F_A^2 - 2F_A) \tag{4.42}$$

When $F_A = 0$ the A modules, in effect, disappear and (4.42) reduces to the pairwise correlation of two elements.

The technique of substitution used above is the basis for analyzing systems with more than two elements.

4.4.3 Generalized Decomposition

The extension of substitution to a three-element parallel system is straightforward. The technique is to choose the failure mechanism modularization so that correlations among similar modules of different elements are the joint correlation of those modules, and contain no correlations of a lower scope. A modular breakdown which accompishes this purpose is shown in Fig. 4.12.

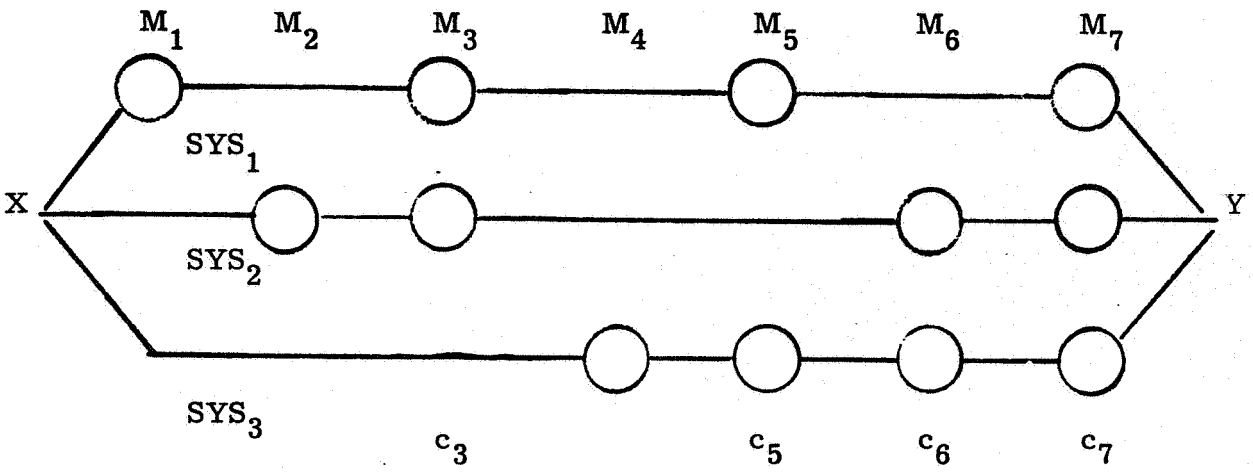


Fig. 4.12

Correlations among the elements manifest themselves as joint correlations of events M_i. For the three-element system, partitioning of the failure mechanisms of each element into seven modules is enough to account for all possible correlation modes (including the mode of no correlation, where a failure mechanism in one element does not appear in the other elements). The c_j represent the degree of correlation in those modules which are common to more than one element.

For example, M_1, M_2 and M_4 represent sets of failure mechanisms which are unique to element 1, element 2, and element 3, respectively; element 1 and element 2 have a common set of failure mechanisms, M_3, which are correlated c_3. The M_i for a given element are mutually exclusive and they exhaust all of the possible failure mechanisms for that element.

The information in Fig. 4.12 can be written as a matrix, where the rows represent redundant system elements and the columns represent sets of failure mechanisms. Each entry in the matrix is the probability of occurrence of a particular failure mechanism set (j) in a particular element (i). The matrix for a three element system is shown in Fig. 4.13.

$$\begin{bmatrix} F_{11} & 0 & F_{13} & 0 & F_{15} & 0 & F_{17} \\ 0 & F_{22} & F_{23} & 0 & 0 & F_{26} & F_{27} \\ 0 & 0 & 0 & F_{34} & F_{35} & F_{36} & F_{37} \end{bmatrix}$$

Fig. 4.13

There is a corresponding vector which gives the correlation, c_j, for each column in the matrix. Note that the non-zero entries in a given column of the matrix need not be equal, i.e., the probability of occurrence of a particular failure mechanism may be different in different elements.

The modular representation of failure mechanisms can be expanded to accomodate N-element systems. The matrix for such an N-element system has dimensions N by 2^N-1.

The first step in computing the probability of system failure is to determine the failure probability of each element. This is

$$F_i = 1 - \prod_{j=1}^{M} (1 - F_{ij}) \tag{4.43}$$

The second step is to combine the failure probabilities of the elements to give the failure probability for the whole system. For an N-element parallel configuration this is just

$$F_{sys} = \prod_{i=1}^{N} F_i$$

$$F_{sys} = \prod_{i=1}^{N} (1 - \prod_{j=1}^{M} (1 - F_{ij})) \tag{4.44}$$

The final step is to include the effects of correlation. The strategy is to consider one term at a time. (F_{sys} is always expressible as a sum of products.) In each term replace the set of factors from each column j with the following expression.

$$c_j F^*_{ij} + (1 - c_{ij}) \prod_{i=1}^{N} (F_{ij} + \delta(F_{ij})) \qquad (4.45)$$

where $\delta(F_{ij}) = 0$, if F_{ij} is present in the term
$\delta(F_{ij}) = 1$, if F_{ij} is not present in the term
F^*_{ij} is the smallest F_{ij} from column j

The method of decomposition and substitution works well in conjunction with the system organization presented Chapter 3. The approach to modularization is by the various system aspects: hardware, software, procedure, etc. The failure probability of a system element is written as a series representation of the failure probabilities of each aspect of the system element. The failure mechanisms associated with each aspect are divided into submodules, as in Fig. 4.12. If there are n elements in the system the number of submodules per system aspect is 2^n-1. Predicting the probability of failure for each system aspect is certainly no more difficult than predicting the probability of failure for the entire system. Predicting correlations for each particular system aspect is probably easier than predicting correlations for the system as a whole.

4.5 More Complex System Configurations

The discussion to this point has centered on active parallel redundancy, because it is widely used and because the algebra is simple enough not to cloud the concepts. The other basic redundancy techniques discussed in Sec. 3.6 are N-modular redundancy (voting), and passive parallelism (dormant backup). Hybrid schemes which employ an N-element voting core supplemented by active or dormant backups are also possible.

4.5.1 N-Modular Redundancy

NMR is a generalization of the TMR voting scheme. For the system to work, it is necessary that a majority of the N system elements be working. NMR is similar to active parallel in that all elements are normally active. It does not make as efficient use of the redundancy, however, as the system fails when just over half of the elements fail. The advantage of this configuration is that it affords rapid and almost transparent recovery from malfunctions. (But, it is also more susceptible to systematic failures because the algorithms tend to be tightly synchronized.)

The joint probability distribution for a three-element system has eight terms. Where system failure in a parallel configuration can be expressed with only one of these terms, a TMR configuration requires four.

$$F_{sys} = F_1F_2F_3 + \bar{F}_1F_2F_3 + F_1\bar{F}_2F_3 + F_1F_2\bar{F}_3 \tag{4.46}$$

When all three elements have the same failure probability the last three terms are equal by symmetry, so for independent elements

$$F_{sys} = F^3 + (1-F)F^2 = 3F^2 - 2F^3 \tag{4.47}$$

The problem of assigning the joint probability so as to include correlations (Sec. 4.3) is the same as in the parallel case: it is difficult to reconcile the various possible correlations. However, some interesting observations can be made if only one of the four possible correlations is present, i.e., correlation is joint over a single subset of elements. This is done by comparing F_{sys} for a correlation over two elements and F_{sys} for a correlation over three elements.

Consider first a three-way correlation. Using the analysis of Sec. 4.3.1 the probability of three elements failing is

$$P_F = cF + (1-c)F^3 \tag{4.48}$$

and the probability of a particular two elements failing is

$$P_F = (1-c)F^2(1-F) \tag{4.49}$$

Substituting (4.48) and (4.49) into (4.46) gives

$$F_{sys} = cF + (1-c)F^3 + 3(1-c)F^2(1-F)$$

$$F_{sys} = cF + (1-c)(3F^2 - 2F^3) \tag{4.50}$$

This makes intuitive sense. It says that an uncorrelated system has a failure probability equal to the theoretical TMR value, and that a totally correlated system has a failure probability equal to that of a simplex element.

Now suppose that two elements, S_1 and S_2, are correlated to each other but not to S_3. The probability of all three elements failing is

$$P_F = cF^2 + (1-c)F^3 \tag{4.51}$$

The probability of only S_1 and S_2 failing varies from $F^2(1-F)$ in the uncorrelated case to $F(1-F)$ in the correlated case, so

$$P_F = cF(1-F) + (1-c)F^2(1-F) \tag{4.52}$$

The remaining two terms, where S_1 and S_2 behave differently, are symmetric. In the uncorrelated case each term equals $F^2(1-F)$. In the correlated case, since S_1 cannot fail while S_2 succeeds, and vice versa, the probability for each term drops to zero, so

$$P_F = (1-c)F^2(1-F) \tag{4.53}$$

Substituting these values into (4.46) gives

$$F_{sys} = cF + (1-c)(3F^2 - 2F^3) \tag{4.54}$$

which is the same as (4.50).

This non-intuitive result says that a TMR system where two of the elements are correlated behaves, statistically, in the same way as a TMR system where all three elements are jointly correlated.

The question arises as to why a TMR system would have pairwise correlations. First of all, even with identical hardware running identical algorithms, hardware production can correlate two elements and not a third. Components from different batches, or assembly by different technicians, can account for such a case. Secondly, there is no requirement that a TMR configuration be run in tight synchronization, or even that the elements be identical. Voting schemes need not be restricted to bit-level-synchronized hardware.

What this all indicates is that TMR (and by extension NMR) systems exhibit a relatively worse reaction to the correlations of smaller scope than do parallel systems.

Another well-known but nonetheless interesting phenomenon associated with TMR systems is that after a certain time they become less reliable than the equivalent simplex system. This time depends upon the reliability function of the simplex system, and for exponential reliabilities equals 0.69 times the Mean Time To Failure. At this point the simplex reliability has fallen to 0.5, and the probability of getting a wrong answer is greater by voting than by just randomly picking the output of one of the simplex elements. Correlations have the strange effect of improving the TMR system performance after this crossover point has been reached. Consider the reliability improvement relation $R = E(c)I(R_u) + R_u$, where $I(R_u)$ is defined by equation (4.4) as $g(R_u) - R_u$. For $R_u > 0.5$, $I(R_u)$ is positive; but for $R_u < 0.5$, $I(R_u)$ becomes negative; $E(c)$ is always a positive fraction. So just as correlation reduces the good effects of redundancy when $I(R_u) > 0$, it reduces the bad effects of the TMR configuration when $I(R_u) < 0$.

The failure mechanism oriented analysis can be applied to TMR systems about as easily as to parallel systems. First compute an expression

for the failure probability of each element F_i by considering the series reliability of the modules in element i, (4.43). Next, substitute these F_i into (4.46) to get F_{sys} where all elements are independent. Finally, compensate for correlation in each term of F_{sys} using expression (4.45).

4.5.2 Passive Parallel Systems

The reliability expressions for systems with dormant backup elements are much more complicated than the expressions for systems where all elements are active. However, it is possible to produce a reliability expression which is a function of both the active and dormant reliabilities of the constituent elements.[2,7] Once this theoretical expression is obtained, the appropriate products of reliabilities (or failure probabilities) can be replaced by the expression (4.45).

Some of the numerical manipulation may seem odd, since in these expressions the reliability of a system element may appear as two different numbers: one for its active condition and one for its passive condition. Two sets of submodularizations may be necessary to reflect the different states of the element. However, this is a straightforward extension of the techniques derived for other redundancy configurations.

4.6 An Example

Even small correlations can have large effects on the improvement in system reliability which is gained with a redundant configuration. These effects can be seen by comparing the ratio reliability improvement measure of a system without correlations, to that of the same system with correlations. The ratio of the two measures is

$$y'/y = (F_u/F)/(F_u/F_r) = F_r/F \qquad (4.55)$$

This is just the ratio of the theoretical system failure probability (figured without correlation) to the real system failure probability (figured with correlation). The correlation is assumed to be joint over all elements.

The results are computed in Table 4.1 for NMR systems of 3, 5, and 7 elements, and for parallel systems of from 2 through 7 elements. 'C' is the correlation and 'F' is the failure probability of a system element.

Two trends are apparent:

1) The effects of the joint correlation become more pronounced as the number of system elements increases. So, the more elements that are needed to achieve a given system reliability, the more detrimental a joint correlation becomes.

2) The effects of a correlation become more pronounced as the ratio of the correlation to the failure probability of an element increases. So, a system composed of highly reliable elements is liable to exhibit a relatively worse reaction to correlations than a system composed of less reliable elements. This can occur because the design procedures that reduce the failure probability of an element do not necessarily reduce correlations due to the residual failure mechanisms in the element. That is, the residual failure mechanisms are likely to correlate to the same extent as those failure mechanisms that were designed out. Since the failure probability has been reduced, the ratio of c to F has increased. Therefore, the effect of the remaining correlations has increased.

C	F	TMR	5MR	7MR
0.500	0.1000	0.4375000000	0.1577008106	0.0312053945
0.500	0.0100	0.0578753156	0.0019681812	0.0000391064
0.500	0.0010	0.0059780777	0.0000199698	0.0000000399
0.500	0.0001	0.0005997801	0.0000002000	0.0000000000
0.100	0.1000	0.7954545455	0.4835065522	0.1387126417
0.100	0.0100	0.2349787100	0.0097640365	0.0001955015
0.100	0.0010	0.0291923325	0.0000998411	0.0000001996
0.100	0.0001	0.0029917229	0.0000009998	0.0000000002
0.050	0.1000	0.8860759494	0.6518428267	0.2436306344
0.050	0.0100	0.3805388839	0.0193392440	0.0003909266
0.050	0.0010	0.0567286241	0.0001996622	0.0000003991
0.050	0.0001	0.0059655984	0.0000019997	0.0000000004
0.010	0.1000	0.9749303621	0.9034872921	0.6169355623
0.010	0.0100	0.7543921827	0.0897531850	0.0019515811
0.010	0.0010	0.2311840975	0.0009975145	0.0000019955
0.010	0.0001	0.0291328131	0.0000099984	0.0000000020
0.005	0.1000	0.9873060649	0.9492968993	0.7630923294
0.005	0.0100	0.8600040403	0.1647220421	0.0038955597
0.005	0.0010	0.3755475691	0.0019930410	0.0000039910
0.005	0.0001	0.0566162360	0.0000199966	0.0000000040
0.001	0.1000	0.9974351667	0.9894306613	0.9415384798
0.001	0.0100	0.9684694932	0.4964832629	0.0191789477
0.001	0.0010	0.7504376719	0.0098863891	0.0000199546
0.001	0.0001	0.2308106575	0.0000999750	0.0000000200

Reliability Improvement with Correlation/Reliability Improvement without Correlation

Table 4.1

C	F	2PAR	3PAR	4PAR
0.500	0.1000	0.1818181818	0.0198019802	0.0019980020
0.500	0.0100	0.0198019802	0.0001999800	0.0000020000
0.500	0.0010	0.0019980020	0.0000020000	0.0000000020
0.500	0.0001	0.0001999800	0.0000000200	0.0000000000
0.100	0.1000	0.5263157895	0.0917431193	0.0099108028
0.100	0.0100	0.0917431193	0.0009991008	0.0000099999
0.100	0.0010	0.0099108028	0.0000099999	0.0000000100
0.100	0.0001	0.0009991008	0.0000001000	0.0000000000
0.050	0.1000	0.6896551724	0.1680672269	0.0196270854
0.050	0.0100	0.1680672269	0.0019962072	0.0000199996
0.050	0.0010	0.0196270854	0.0000199996	0.0000000200
0.050	0.0001	0.0019962072	0.0000002000	0.0000000000
0.010	0.1000	0.9174311927	0.5025125628	0.0909918107
0.010	0.0100	0.5025125628	0.0099019705	0.0000999901
0.010	0.0010	0.0909918107	0.0000999901	0.0000001000
0.010	0.0001	0.0099019705	0.0000010000	0.0000000001
0.005	0.1000	0.9569377990	0.6688963211	0.1668056714
0.005	0.0100	0.6688963211	0.0196097657	0.0001999602
0.005	0.0010	0.1668056714	0.0001999602	0.0000002000
0.005	0.0001	0.0196097657	0.0000020000	0.0000000002
0.001	0.1000	0.9910802775	0.9099181074	0.5002501251
0.001	0.0100	0.9099181074	0.0909173561	0.0009990020
0.001	0.0010	0.5002501251	0.0009990020	0.0000010000
0.001	0.0001	0.0909173561	0.0000099999	0.0000000010

Reliability Improvement with Correlation/Reliability Improvement without Correlation

Table 4.1

C	F	5PAR	6PAR	7PAR
0.500	0.1000	0.0001999800	0.0000199998	0.0000020000
0.500	0.0100	0.0000000200	0.0000000002	0.0000000000
0.500	0.0010	0.0000000000	0.0000000000	0.0000000000
0.500	0.0001	0.0000000000	0.0000000000	0.0000000000
0.100	0.1000	0.0009991008	0.0000999910	0.0000099999
0.100	0.0100	0.0000001000	0.0000000010	0.0000000000
0.100	0.0010	0.0000000000	0.0000000000	0.0000000000
0.100	0.0001	0.0000000000	0.0000000000	0.0000000000
0.050	0.1000	0.0019962072	0.0001999620	0.0000199996
0.050	0.0100	0.0000002000	0.0000000020	0.0000000000
0.050	0.0010	0.0000000000	0.0000000000	0.0000000000
0.050	0.0001	0.0000000000	0.0000000000	0.0000000000
0.010	0.1000	0.0099019705	0.0009990110	0.0000999901
0.010	0.0100	0.0000010000	0.0000000100	0.0000000001
0.010	0.0010	0.0000000001	0.0000000000	0.0000000000
0.010	0.0001	0.0000000000	0.0000000000	0.0000000000
0.005	0.1000	0.0196097657	0.0019960279	0.0001999602
0.005	0.0100	0.0000020000	0.0000000200	0.0000000002
0.005	0.0010	0.0000000002	0.0000000000	0.0000000000
0.005	0.0001	0.0000000000	0.0000000000	0.0000000000
0.001	0.1000	0.0909173561	0.0099010881	0.0009990020
0.001	0.0100	0.0000099999	0.0000001000	0.0000000010
0.001	0.0010	0.0000000010	0.0000000000	0.0000000000
0.001	0.0001	0.0000000000	0.0000000000	0.0000000000

Reliability Improvement with Correlation/Reliability Improvement without Correlation

Table 4.1

CHAPTER 5

CONCLUSIONS

5.1 Seriousness of the Problem

Systematic failures, or more properly, correlated malfunctions, are acknowledged to exist in redundant system configurations, but they are generally overlooked in reliability analyses. Perhaps this is because such analyses are usually concerned with just the physical hardware, not the whole system, and because the correlations between redundant modules of physical hardware are considered to be too small to worry about.

Neither of these reasons for neglecting the effects of correlations is particularly valid. The probable explanation for their persistence is that, to date, no highly redundant ultra-reliable systems have been built (and observed for a prolonged period of time). In the redundant systems that have been observed, correlated failures, if they are recognized as such, are treated as freak occurrences. Nevertheless, there is sufficient evidence of the existence of systematic failure modes in hardware to warrant that the problem be treated thoroughly.[1,5,9]

Neglecting to consider the imperfection of system algorithms in a reliability analysis is almost inexcusable. Experience with the Apollo Guidance and Navigation System indicates that the majority of operational difficulties are due to algorithmic flaws (procedures, software, and hardware function). Algorithms correlate in a more obvious manner than hardware, as often the "same" algorithm is used on redundant hardware elements. In such a case, the coefficient for algorithmic correlation approaches one. This is clearly non-negligible.

Section 4.6 shows that correlations as small as .001 can virtually negate the benefits of redundancy. As ultra-reliable systems employ more

redudancy, and as system elements become more reliable, correlated failure modes will have an increasingly noticeable effect on overall system performance. It is incumbent upon the practioners of the art of fault-tolerant system design to include the effects of correlation in their analyses.

5.2 Predicting Values for Correlation

Accurate prediction of the values of the correlations in a system is not an easy job. Potential correlations can be extracted from the System Development Structure of Chapter 3. Analysis of the structure will provide some information about the relative magnitude of the various correlations.

Engineering judgement and a certain amount of guesswork must be combined with any available statistics about the occurrence of similar "potential correlations" in other systems in order to predict an absolute magnitude for correlations in a system under consideration. As a better understanding of the system failure mechanisms develops, the more accurate will become the prediction of correlation values.

Even though accurate prediction of the value of correlations is still a major difficulty, even a deliberate underestimation of correlations provides a more accurate picture of system reliability than is now available.

5.3 Comparison of Mathematical Approaches

Three ways of including correlations in reliability equations have been considered.

* Section 4.2 discusses the use of effectiveness factors, E(c) and X(c), on reliability improvement measures.

* Section 4.3 discusses correlations from a system point of view. Each redundant element in a system is taken to be a black box with failure probability F_i. The correlations among all different combinations

of elements are then factored into the reliability equation, without regard for which aspects of the elements correlate.

* Section 4.4 considers the failure mechanisms of each system element. The failure probability of an element is factored into the failure probabilities of selected groups of failure mechanisms, called failure mechanism modules. These modules can be correlated in a straightforward manner.

The effectiveness factor approach has by far the simplest equations. The difficulty is that it reduces all correlation coefficients to a single number. For all but symmetrical cases (joint correlation over all system elements) the equations are inexact. This is not as serious a shortcoming as it may seem. A great many practical systems are symmetric by design, and any asymmetries which arise during development will produce effects of second order. In any case, the inaccuracies in the knowledge of the F_i's and the correlations often makes this a reasonable approximation.

The second method attempts an exact approach to the problem. For the general case, with asymmetric correlations, it is difficult to determine the reliability equation. However, the equation for the joint correlation of N elements derived by this method is essential for the approach in Sec. 4.4.

The failure mechanism approach of Sec. 4.4 also provides an exact answer, but the equation for system reliability is easier to generate than in the system oriented approach. The method requires that the system elements be broken down into failure mechanisms. This is not as much of a problem as it may seem, for a good deal of the work is done by determining the correlation relations in the first place. The degree of decomposition should equal the level to which correlations are determineable. It is unprofitable to decompose more than this as no additional information is put into the reliability equation. A fairly natural decomposition is that found in the system development structure of Chapter 3. Standard techniques

are available for finding hardware reliability; techniques for evaluating software reliability are corrently under investigation. The methods of software evaluation proposed Shooman[10] can also be applied to other levels of algorithm.

The effectiveness factor approach and the failure mechanism decomposition are two usable methods for putting the effects of correlations into system reliability equations. The choice depends upon the amount and accuracy of available information about the system and system development.

5.4 Suggestions for Future Work

This study has just broken the surface of the correlated malfunction problem. In order for the correlation phenomenon be incorporated into the current work on fault-tolerant system analysis, the techniques for including correlations in the reliability equation must be refined and the effects of correlations must be investigated in greater depth. Specifically:

1) The equations of Sec. 4.4 can be expanded for various types of hybrid systems with dormant backups.

2) The System Development Structure can be refined to further assist in determining the existence of correlations and in determining their values.

3) Further computations of redundancy effectiveness can be done using the reliability equations of Sec. 4.4 and more sophisticated reliability improvement measures.

REFERENCES

1. Aviation Week and Space Technology, January 4, 1971 p. 17, January 11, 1971 pp. 21-22, January 25, 1971 p 15.

2. W. G. Bouricius, W. C. Carter, et. al., Reliability Modeling Techniques for Fault-Tolerant Computers, IEEE Transactions on Computers, Volume C-20, pp. 1306-1311, November 1971.

3. E. W. Dijkstra, Notes on Structured Programming, Technological University of Eindhoven, Publication EWD249, June 1971.

4. E. W. Dijkstra, A Constructive Approach to the Problem of Program Correctness, BIT Volume 8, pp. 174-186, 1968.

5. E. C. Hall, Reliability History of the Apollo Guidance Computer, MIT/C. S. Draper Lab Report R-713, January 1972.

6. T. F. Klaschka, A Method for Redundancy Scheme Performance Assessment, IEEE Transactions on Computers, Volume C-20, pp. 1371-1376, November 1971.

7. F. P. Mathur, On Reliability Modeling and Analysis of Ultrareliable Fault-Tolerant Digital Systems, IEEE Transactions on Computers, Volume C-20, pp. 1376-1382, November 1971.

8. P. G. Neumann, A Hierarchical Framework for Fault-Tolerant Computing Systems, Digest of Papers of the Sixth Annual IEEE Computer Society International Conference, pp. 337-340, September 1972.

9. J. Partridge, L. D. Hanley and A. C. Metzger, Reliability for the LSI Age, MIT/C. S. Draper Lab Report E-2370, December 1968.

10. M. L. Shooman, Probabilistic Models for Software Reliability Prediction, Digest of Papers of the 1972 International Symposium on Fault-Tolerant Computing, pp. 211-213, June 1972.

11. K. Vandivier, "The Great Aircraft Brake Scandal", Harper's Magazine, pp. 45-53, April 1972.

12. J. von Neumann, "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components", Automata Studies, Annals of Mathmatics, Number 34, pp. 43-98, Princeton, 1956.

13. W. B. Davenport, Probability and Random Processes, McGraw-Hill, New York, 1970.

14. M. L. Shooman, Probabilistic Reliability, McGraw-Hill, New York, 1968.